(2)

MASSACHUSETTS INTITUTE OF TECHNOLOGY VLSI PUBLICATIONS

COMPUTER-AIDED FABRICATION OF INTEGRATED CIRCUITS

Semiannual Technical Report for the period April 1, 1989 to September 30, 1989

Massachusetts Institute of Technology
Cambridge, Massachusetts 02139

Principal Investigators:	Paul Penfield, Jr.	(617) 253-2506
	Dimitri A. Antoniadis	(617) 253-4693
	Stanley B. Gershwin	(617) 253-2149
	Stephen D. Senturia	(617) 253-6869
	Emanuel M. Sachs	(617) 253-5381
	Donald E. Troxel	(617) 253-2570

This research was sponsored by the Defense Advanced Research Projects Agency (DoD), through the Office of Naval Research under Contract No. MDA972-88-K-0008. Monitored by John C. Toole.

Microsystems Technology Laboratories
Massachusetts Institute of Technology
Cambridge Massachusetts 02139
Room 39-321
Telephone
(617) 253-0292

TABLE OF CONTENTS

Selected Publications (page 10):

Emanuel Sachs, Ruey-Shan Guo, Sungdo Ha, and Albert K. Hu, "Process Control System for VLSI Fabrication," to be published in the Proceedings of the Fall 1989 Electrochemical Society Meeting, October 1989.

Michael P. Ruf, "Retrieving and Integrating IC Fabrication Data from Dissimilar Databases," VLSI Memo No. 89-552, MIT, June 1989.

X. Bai and S. B. Gershwin, "A Manufacturing Scheduler's Perspective on Semiconductor Fabrication," VLSI Memo No. 89-518, MIT, April 1989.

RESEARCH OVERVIEW

The major objective of this research program is to develop a Computer Aided Fabrication (CAF) environment to support the design, implementation, and manufacturing of processes for integrated circuits with special emphasis on prototyping and application specificity. The primary testbed for all components of the CAF system is the Integrated Circuits Laboratory at MIT. These components are the result of research in five areas under the direction of the individuals listed below:

- CAF System Architecture (Professor Donald E. Troxel)
- Modular Process Design (Professor Dimitri A. Antoniadis)
- Scheduling of Fabrication Events (Dr. Stanley B. Gershwin)
- Equipment Modeling (Professor Emanuel M. Sachs)
- Mechanical Property TCAD (Professor Stephen D. Senturia)

The remainder of this report discusses progress made in each of these areas. Further information is contained in the publications listed in the last section.

CAF SYSTEM ARCHITECTURE

Mike Heytens and Jeff Thompson developed substantial new software to permit changes in the CAFE schema by applications programmers. Previously, such changes had to be performed by our data base guru. Now applications programmers can experiment with new schema definitions and test them by using the new schema in their applications programs. This facilitates a much more rapid development of applications programs and reduces the dependence on our data base guru.

Mike Heytens and Mike McIlrath have developed an interface to our Gestalt data base system through the Common Lisp Object System (CLOS). The interface provides transparent access to persistent objects, which are described and manipulated solely via CLOS constructs. This transparency will simplify applications programming, as it frees the programmer from the task of translating between programming language structures and data base structures, which is required in many systems. Applications programmers may utilize the rich object modeling, generic functions, interactive debuggers, interpreters, etc. present in the CLOS environment to further simplify programming.

There has been substantial interaction between Stan Gershwin, his students, Don Troxel, and Abbas Kashani. This has resulted in the definition of a "low level" scheduler which is now the S. M. thesis project of Kashani. This project is designed to accept the output of the higher level scheduler to be written by Stan Gershwin's students and to aid users of our Integrated Circuits Laboratory (ICL) in actually making reservations for machines and people to operate these machines. In conjunction with this project, the CAFE schema has been modified to provide for a people resource as well as a machine resource. In addition, several enhancements to the present reservations program have been made, many of which were in response to user comments and requests.

Our Process Flow Representation (PFR) has been refined and extended. Duane Boning and Mike McIlrath have collaborated to extend the PFR and to integrate it and the associated core evaluator, which is used to evaluate the PFR and return objects of interest for further processing by other interpreters, into the simulation management software developed by Duane Boning.

Duane Boning has produced a rather remarkable system for the simulation of IC processing from our PFR. Users can enter and/or edit the process flow representation directly in its LISP-like syntax or with the forms based editor previously developed by Rajeev Jayavant. They can then run SUPREM III simulations without having to know anything about the details of SUPREM III. A key feature of this software is the "validator" which checks the validity of previous simulation computations after the process flow representation has been modified and retains computations which are still valid. Thus computation time is minimized while maintaining correctness of the final simulations. This software also allows the user to conveniently generate reports, such as plots of impurity concentrations and calculations of sheet resistance, etc.

This past summer, we made a major update to the CAFE system running in the ICL fabrication facility. This included an update to the Sun 4 version of the operating system, new schema, enhancements to several applications programs, and a switch to Kyoto Common Lisp (KCL). One enhancement to the application programs is Fabform user interface, which allows individual applications programs to specify the form to be used, describe the fields and their characteristics, including possible default values and/or restricted choices, and also handles user interaction. Other enhancements included equipment reservations, operate machine, wip and lot tracking, and new utilities for creating lots and splitting lots into multiple wafer sets.

KCL is in the public domain, which removes one more licensing hurdle when we distribute CAFE to another site. We have also brought up KCL on Vaxes as well as the Suns. The switch to

KCL provided a significant increase in performance due in part to increased shared text space of approximately three megabytes. The swap space startup overhead is now less than five megabytes per invocation of the CAFE program after the first one is running. This contrasts with the 10-11 megabytes for our earlier Allegro implementation. The startup time is now approximately four to five seconds for all but the first CAFE.

We have installed a new Lisp-based menu system which allows users to customize menus to their liking simply by executing some Lisp functions. Support functions are provided to define new menus, add a binding to a menu, delete a binding from a menu, retrieve status of current menu definitions, etc. The menus may be modified dynamically even after the menu system is running. Functionality is similar to the previous menu system from the user's standpoint. Much more extensive support is provided for Lisp functions bound to menu items or header fields. Full documentation is provided on line in /usr/cafe/doc.

Several new features have been added to Fabform since the last release. We now have the ability to pass input parameter and template "files" in strings rather than through disk files. The new menu system uses this feature to avoid generating hundreds of temporary files.

The forms-based-flow editor is included with this release of CAFE. It can be used to build flows from predefined operations or build new operations. In the future it will be enhanced to allow editing of the PFR.

We have the ability to build a "minicafe" which consists of CAFE without Gestalt or Gestalt-dependent tools. The flow editor has been modified to look for the existence of Gestalt and only rely on Gestalt if it is available.

MODULAR PROCESS DESIGN

Work on the Profile Interchange Format (PIF) has been directed primarily toward implementation of an "intertool" format, which is best described as an application programming interface (API) to PIF information for direct use in process and device CAD programs. This is in strong distinction to an "intersite" format, which is a textual definition or ASCII file format for the storage and transmission of device geometry and attribute information. Ph.D. candidate, Duane Boning has put in place an implementation of the intersite PIF layered on the Gestalt database interface; we refer to this implementation as the "PIFLIB". Based on experiments with the PIFLIB, we found that extensions to Gestalt were needed, including better support for array information, and more complete handling of class hierarchies to support object-oriented programming. These extensions were carried out by Mike Heytens. Also based on our experience, we have begun collaborative efforts with Professor Andrew Neureuther and Alexander Wong of UC Berkeley (where work with PIF on the OCT database is underway) to define a "standard" PIF API. We are now working to implement this PIF API using Gestalt.

The Technology (encompassing both Process and Device) CAD Environment is based extensively on the wafer representation (PIF) and the process flow representation (PFR). The intent is to provide an environment suitable for the integration and use of existing TCAD tools, as well as to investigate new kinds of tools for technology development. We have made progress on two components of the TCAD environment. First, the Simulation Manager has been rewritten to enable the designer to better relate simulation results to the Process Flow Representation, and to provide for the control of different kinds of simulations from the Process Flow Representation. Our goal is for the Simulation Manager to support treatment-level simulation (via Suprem-III, and in the future Suprem-IV), change-in-wafer-state level simulation (via Simpl-2), and mechanical simulation (to support Mechanical Property TCAD research).

The second component of the TCAD Environment that has seen substantial progress is the Process Advisors. This was the subject of a Bachelor's Thesis by Partha Saha. The Process Advisors provide direct estimates of unknown treatment parameters given goal level information (a "process synthesis" capability). We have extended the Advisors to use previous simulation results as a starting point, to provide analytic models as a fast alternative to simulation for exploration of process parameter effects, and to solve for additional unknowns in some cases (i.e., for either diffusion, temperature, or time). X11 interfaces to the oxidation, diffusion, and implantation unit process advisors have been implemented, and the advisors released for experimental use as the "AdvisorTool". An implementation using DEC Windows in the VAX environment at Digital's plant in Hudson, Massachusetts has been implemented by Duane Boning.

SCHEDULING OF FABRICATION EVENTS

Research on the following activities continued in this period:

(a) Systems level study of the integrated circuits fabrication process.

We have identified many of the different events that affect scheduling in an integrated processing facility and we have identified appropriate decisions in response to those events. The machine and human resources that need to be scheduled have also been identified.

(b) Formulating a mathematical model of an integrated circuits fabrication facility.

The basic model is being enhanced to include more phenomena like non-zero storage capacity between production stages, scheduling of batch machines and scheduling machines with a significant changeover time from one operation to another (such as etchers and ion implanters.)

Xiewei Bai has developed a decomposition technique that greatly reduces the computational complexity of the scheduling algorithm by decoupling the decision making in different portions of the factory during periods of time when they operate independently.

Romuald De Sigy is developing algorithms for scheduling batch machines. When a particular lot of wafers arrives at a batch machine, a decision has to be made about the time at which the lot will be loaded into the machine. A numerical solution has been obtained for this problem. Heuristics are being developed for the case where there are a variety of wafers.

Srivatsan is studying the effects of scheduling policies on performance when the machines have a significant changeover time. A mathematical model has been developed which gives the frequencies with which machines change from one operation to another. This information is used in developing policies for scheduling setup changes.

(c) Developing simulation and scheduling software.

A simulation package has been developed (primarily by Bovornrat Darakananda) which implements the hierarchical control algorithm. A number of simple systems have been simulated by Darakananda, James Violette, and others. Bai has simulated the fabrication flow for the baseline CMOS process of the MIT Integrated Circuits Laboratory (ICL) and has obtained results which are remarkably close to what is actually obtained in practice.

A real time controller for scheduling the Low Temperature Oxide LPCVD machine in ICL has been developed.

Other fabrication facilities under study include the IBM East Fishkill facility for manufacturing chips and a Texas Instruments facility for making linear integrated chips.

EQUIPMENT MODELING

During the past six months, work has continued on two projects: the development of a complete system for process control and the formalization of the use of dimensional analysis in statistical modeling.

Doctoral Candidate Ruey-Shan Guo has made considerable progress in further definition of our complete process control system. This system is described in detail in an accompanying paper which will appear in the proceedings of the Fall 1989 Electrochemical Society Meeting. It is a modular system which accomplishes unit process control with consideration of preceding and following process steps. The basic control function has been divided into three modules: the "flexible recipe generator", the "run by run controller", and the "real time controller". The "flexible recipe generator" serves to create a recipe in response to a new ASIC process design. The role of this module is to take the best guess it can about where to begin operations. The "run by run controller" serves to update the recipe on a run by run basis. That is, it modifies the recipe in time for the next run but not during a current run. The "real time controller" modifies the recipe during a process step in response to in-situ measurements. The division of the control function into three core modules is a division of the control function according to frequency of corrective action and the size of the control space covered. The flexible recipe generator will be used once per new design, but will cover a large process parameter space. The run by run controller will be invoked once between each batch and will cover a smaller range of space than the flexible recipe generator. The real time controller will be used frequently in the course of a batch, but will span a smaller range of operating space than the run by run controller.

In addition to the general planning of the control system, work has proceeded on the implementation of the run by run controller. The run by run controller is built on the framework established by a product called "Ultramax". Ultramax implements sequential design of experiments in order to optimize a process without making scrap. Ultramax has been applied to the mechanistic model for the LPCVD of polysilicon which was developed previously in this program by George Prueger, in order to test its efficacy as the basis for a run by run controller. It has been found that it does indeed optimize the uniformity of the growth rate profile down the length of the tube. However, the optimization accomplished by Ultramax is quite slow. This is due to the fact that the model for uniformity is a quadratic polynomial in five variables and therefore has 21 coefficients that must be calibrated from the data. It therefore takes at least 21 runs to optimize the process.

At the present time, we are working on a methodology for dramatically increasing the speed of optimization and control using the sequential design of experiments. The basis of the methodology is to model directly the measured parameter, creating one model for each measurement site. Then, the quality metric, such as uniformity, can be calculated from these models. This a substitute for the current practice of creating one model for the quality metric itself. The advantage of the new approach is that in the region of the optimum for the quality metric, the individual measurements are not near an optimum. Thus, while the quality metric must be modeled by a high order expression like a quadratic, the direct measurements can be modeled by simple linear expressions. The result is a much faster optimization as there are fewer terms to calibrate from the data.

Master's candidate Ka Shun Wong has picked up the work on the use of Dimensional Analysis that was started by Master's student William Wehrle. Wehrle was able to show by specific example, that the use of Dimensional Analysis to group variables was able to create models with higher accuracy for a given number of terms than primitive models. Mr. Wong is developing a general method for the use of Dimensional analysis, which can be easily applied to new processes.

MECHANICAL-PROPERTY TCAD

Effort during this reporting period was concentrated on three tasks: (a) completion of the acquisition, installation, and preliminary training on and use of mechanical CAD tools; (b) exercising of the mechanical CAD system for 3-D solid modeling and mechanical finite-element modeling of microelectronic structures and micromechanical structures; and (c) literature research and compilation of the first entries in the mechanical property database which is an essential part of the mechanical CAD system. Specific activities in each area are reported below.

(a) CAD system

The mechanical CAD tool PATRAN and the finite-element program ABAQUS were installed on our Sun 4, along with the PATRAN linking program between the two codes. As a result, it is now possible to use the PATRAN interface to create 3-D solid models, set up finite-element meshes, boundary conditions, and loads, transport the modeling problem to ABAQUS, receive the ABAQUS output file, and display both numerical and graphical results from the modeling.

Fariborz Maseeh, the graduate student on this program, took a training course both in PATRAN and ABAQUS, and is preparing to teach an internal seminar on their use to other students (presently planned for January 1990).

There remains a need for getting a version of PATRAN that supports X11 window display protocol. This has been promised by the vendor for November 1989.

(b) 3-D solid modeling and finite-element modeling

Together with two undergraduate summer students (Miles Arnone and Sean Gelston), Maseeh exercised the 3-D modeling capability of PATRAN by creating a model of a CMOS cell conforming to the dimensions of the MIT baseline CMOS process. One result of this effort was the identification of several residual bugs in the PATRAN graphics processor. The vendor promises that the November 1989 version will fix these bugs.

The complete PATRAN-ABAQUS link has been used to model deformable structures, which are critical for the measurement and confirmation of mechanical properties of microelectronic materials. Specifically, the static load-deflection behavior of square and circular suspended membranes with residual tensile stress, and undergoing large deflections, has been successfully modeled, and is being linked with other experimental programs on mechanical property measurement. In addition, the resonance behavior of a cantilever beam with a variously-positioned etched hole has been modeled, in anticipation of a series of experiments on this structure planned for the Fall of 1989.

(c) Mechanical Property Database

The two summer students (Arnone and Gelston) carried out an extensive literature search on the mechanical properties of silicon nitride and silicon dioxide. Several hundred papers were examined, and relevant experimental results were gleaned, collated, and compiled into reports that will be issued during October 1989. In addition, results from these surveys were entered into the PATRAN neutral file, which has been selected as the first implementation of the mechanical property database.

PUBLICATIONS LIST

Parmeet Singh Chaddha, "*Comparison of equipment Modeling Methods as Applied to the LPCVD of In-Situ P-Doped Polysilicon,*" M.S. Thesis, MIT, Department of Mechanical Engineering , Cambridge, Massachusetts, March 1989; also MIT VLSI Memo No. 89-517, March 1989.

Michael P. Ruf, "*DRIFS - A Data Retrieval Interface for Integrated Circuit Fabrication Systems,*" M.S. Thesis, MIT, Department of Electrical Engineering and Computer Science, Cambridge, Massachusetts, April 1989; also MIT VLSI Memo No. 89-520, April 1989.

Michele E. Storm, "*Sequential Design of Experiments with Physically Based Models,*" M.S. Thesis, MIT, Department of Mechanical Engineering, Cambridge, Massachusetts, April 1989; also MIT VLSI Memo No. 89-529, May 1989.

William Paul Wehrle, "*Application of Dimensional Analysis to Statistical Process Modeling,*" M.S. Thesis, MIT, Department of Mechanical Engineering, Cambridge, Massachusetts, March 1989; also MIT VLSI Memo No. 89-533.

Partha Saha, "*IC Process Synthesis by Analytical Models,*" S.B. Thesis, MIT, Department of Physics, Cambridge, Massachusetts, May 1989.

Fariborz Maseeh, and Stephen D. Senturia "Plastic Deformation of Highly Doped Silicon," presented at the 5th International Conference on Solid-State Sensors and Actuators, Transducers 1989, Montreux, Switzerland, June 1989. To be published in *Sensors and Actuators Proceedings of the IEEE.*

Emanuel Sachs, Ruey-Shan Guo, Sungdo Ha, and Albert K. Hu, "Process Control System for VLSI Fabrication," to be published in the *Proceedings of the Fall 1989 Electrochemical Society Meeting*, Hollywood, Florida, October 16-17, 1989.

INTERNAL MEMORANDA

Xiewei Bai and Stanley B. Gershwin, "A Manufacturing Scheduler's Perspective on Semiconductor Fabrication," VLSI Memo No. 89-518, MIT, Cambridge, Massachusetts, April 1989.

Michael P. Ruf, "Retrieving and Integrating IC Fabrication Data from Dissimilar Databases," VLSI Memo 89-552, MIT, Cambridge, Massachusetts,.June 1989.

TALKS WITHOUT PROCEEDINGS

Stephen D. Senturia, "Mechanical Properties of Microelectronic Materials: Why we care," presented at the Recent Advances in VLSI Symposium, MIT, Cambridge, Massachusetts, April 4, 1989.

Paul Penfield, Jr.,"Computer-Aided Fabrication of Integrated Circuits," presented at the Recent Advances in VLSI Symposium, MIT, Cambridge, Massachusetts, April 4, 1989.

Emanuel Sachs, Albert Hu, Sungdo Ha, and Ruey-Shan Guo, "A Framework for Process Control in VLSI Fabrication," presented at the MIT VLSI Research Review, MIT, Cambridge, Massachusetts, May 22, 1989.

Stanley B. Gershwin and Xiewei Bai, "Real-Time Scheduling of Semiconductor Fabrication Lines," presented at the MIT VLSI Research Review, MIT, Cambridge, Massachusetts, May 22, 1989.

Duane S. Boning and Michael B. McIlrath, "The MIT Process Flow Representation: Definition and Application," presented at the MIT VLSI Research Review, MIT, Cambridge, Massachusetts, May 22, 1989.

Emanuel Sachs, "The Use of Experimental Design in VLSI Process Control," presented at Dupont Corporation Statistical Process Control Group, Wilmington, Delaware, May 23, 1989.

Emanuel Sachs, "Equipment Modeling and Process Control," presented at Applied Materials Corporation, Santa Clara, California, June 15, 1989.

Emanuel Sachs, "Equipment Modeling and Process Control," presented at Intel Corporation, Santa Clara, California, June 15, 1989.

Emanuel Sachs, "Equipment Modeling and Process Control," presented at Rockwell International Science Center, Thousand Oaks, California, June 16, 1989.

Stephen Senturia, "Critical Issues in Microsensor Design," presented at Schlumberger, Montrouge Research Center, France, June 1989.

Stephen Senturia, "Critical Issues in Microsensor Design," presented at IBM T. J. Watson Laboratory, July 1989.

Emanuel Sachs, "The Application of Dimensional Analysis to the Modeling of the LPCVD of Doped Polysilicon," presented at BTU Bruce Corporation, Billerica, Massachusetts, July 20, 1989.

D. S. Boning, "Manufacturing Integration," presented at the SRC Workshop on the Process Engineering Toolbox, Stanford University, California, August 3-4, 1989.

Donald E. Troxel, "Process Control and the CAFE Architecture," presented at the SRC/DARPA Workshop on CIM, University of Michigan, August 1989.

Emanuel Sachs, "VLSI Process Control," presented at the SRC/DARPA Workshop on CIM, University of Michigan, August 1989.

D. S. Boning, "Toward Process Synthesis: Unit Process Advisors," presented as a DEC Technical Forum, Hudson, Massachusetts, September 5, 1989.

Emanuel Sachs, "A Proposal for Process Control of Epitaxy," presented at the SRC Technical Research Conference on Epitaxy, MIT, Cambridge, Massachusetts, September 26, 1989.

PROCESS CONTROL SYSTEM FOR VLSI FABRICATION

Emanuel Sachs, Ruey-Shan Guo, Sungdo Ha, Albert K. Hu
Massachusetts Institute of Technology
Room 35-231, 77 Massachusetts Avenue
Cambridge, Massachusetts 02139

ABSTRACT

A modular framework for the implementation of process control in VLSI fabrication will be described. The system integrates existing approaches to process control with new methodologies in order to achieve on-line optimization and control of unit processes with consideration of preceding and following process steps. The process control is based on three core modules. The Flexible Recipe Generator determines an initial point for operations in response to a new product design. The Run by Run Controller tunes the recipe between runs based on feedback from post process measurements and feedforward measurements. The Real Time Controller further modifies the recipe during a process step based on in-situ measurements. Algorithmic bases of these modules will be described and potential research issues be identified. Optimization and control of LPCVD of polysilicon were achieved by applying the Flexible Recipe Generator and the Run by Run Controller.

I. INTRODUCTION

Process control is a crucial ingredient in the operation of any semiconductor fabrication facility. The techniques and methodologies used for process control determine the effectiveness of each piece of equipment and influence the organization and operation of a facility. Our work concerns the development of a modular system for process control which addresses the increasingly stringent demands for high yield on both large and small lots of wafers. The system integrates existing approaches to process control with new methodologies in order to achieve on-line optimization and control of unit processes with consideration of preceding and following processes.

A. Current Practice

Draft of Paper for ECS. Fall 1989

In current practice, process optimization and control are accomplished at two stages; at the process design stage (off-line) and at the manufacturing stage (on-line). Process optimization is usually done on an off-line basis, while process control on an on-line basis (Figure 1).

The purpose of off-line process optimization is to, at the design stage, determine the parameter settings which optimize the output performance. Response surface method (RSM) (1,2,3) and the Taguchi method (4,5,6,7) are now widely used for off-line process optimization. In both methods, statistical design of experiments are used to help explore the global process space efficiently.

In RSM, regression models which relate outputs to input parameters are often constructed for the estimation of optimum operating point. In the Taguchi method, process variability is included as part of the response. The goal of his parameter design is to determine the settings of some controllable parameters so as to minimize the effects of disturbances on output variation. In our opinion, both methods are invaluable tools and aid in the construction of our process control system.

The purpose of on-line process control is to, at the manufacturing stage, maintain the best output performance. Proportional-integrated-derivative (PID) feedback control and statistical process control (SPC) are now widely used for this purpose.

Most equipment comes with some built in PID process controllers, such as temperature controllers, pressure controllers and mass flow controllers. These feedback control loops are independent, each acting on one equipment parameter. Thus, when equipment parameters are set, we are in fact establishing the set point of one of the feedback control loops resident in the machine.

Once placed on the production floor, the equipment parameters and post process measurements can be measured and plotted on SPC charts such as Shewhart or CUSUM charts (8,9). These methods investigate the variability of the process, distinguish between random and assignable causes, and aid in the quick identification of deviations from routine operation so that corrective action may be undertaken.

B. Opportunities for Improvement

Continuing demands for high yield on both large and small lots of wafers are creating a need to improve the current process control system. New methodologies should be proposed to compensate for the inadequacies of the current system. More specifically, the improved system should have the new features of achieving optimization without making scrap, reducing the output variation as small as possible, and performing the control with more automation.

As described above, off-line optimization serves to explore the global process space for the suggestion of optimum operating point. Since most of the designed experiments produce scrap, off-line optimization is infrequently done, and is used only to determine the best general area for operation and not to fine tune the process. An algorithm which performs further on-line optimization without making scrap will be necessary (Figure 1).

The current PID feedback control on equipment settings is not enough to reduce the output variation which causes yield loss. Further reduction can be accomplished by controls involving small changes in the equipment parameters around the optimum operating point. Feedback control based on post process measurements and in-situ measurements to compensate for the mean shifts and drifts will reduce the output variation. Feedforward control based on post process measurements of a previous step might also contribute to the reduction of the output variation (Figure 1).

Currently, automated SPC interpretation is rarely done and equipment diagnostics is mainly through human interpretation. Greater automation is expected in the new system to help identify problems as early as possible.

II. System Description

A. Block Diagram Description

The modular framework of the process control system under development is schematically illustrated in figure 2. This block diagram represents the control of three sequential pieces of equipment (n-1, n, and n+1), which occur somewhere in the process flow corresponding to the manufacture of an integrated circuit. The square cornered boxes represent hardware, the rounded corner boxes represent software modules, and the ellipses represent information flowing through the system.

Each piece of equipment has post process and in-situ measurements associated with it. In specific instances, these measurements might be absent, but in general they are available. Post process measurements are those made on wafers after they are removed from the processing equipment. Examples of post process measurements include thickness measurements on deposited films, and line width measurements. In-situ measurements are those made at the processing equipment while the process is going on. In-situ measurements can be of three types; measurements made on the wafer itself, measurements made on equipment parameters, and measurements made on process parameters. An example of an in-situ wafer measurements is in-situ ellipsometry for film thickness. An example of an in-situ equipment parameter measurement is a measurement of temperature. An example of an in-situ process parameter measurement is a measurement of gas composition in a reactor chamber.

The function of process control has been divided into three core modules; the "Flexible Recipe Generator", the "Run by Run Controller", and the "Real Time Controller". These three modules act to divide up the control function on the basis of the time scale of the response and the range of operating space encompassed. The Flexible

Recipe Generator covers the largest operating space and is invoked only once at the beginning of manufacture of each new design or product. The Run by Run Controller covers a smaller range of space and is invoked (in the limit) after each run of product. The Real Time Controller has the smallest range of operating space and is invoked (in the limit) continuously during each product run.

In response to a new product design, the Flexible Recipe Generator has the task of taking the best first guess about where to begin operations, resulting in a recipe for use in the first run of a new product. Conceptually, the Flexible Recipe Generator is a substitute for off-line design of experiments and is expected to get the process into the general region of the best performance. Further fine tuning of the process is accomplished by the Run by Run Controller and Real Time Controller modules.

The Run by Run Controller starts from the recipe provided by the Flexible Recipe Generator. It then updates the recipe between product runs based on the results of post process and in-situ measurements from the current process step and post process measurements from previous process steps. The Run by Run Controller serves the multiple purposes of local optimization, feedback control and feedforward control. Local optimization is the process of fine tuning a recipe given a general region in which the optimum lies. Feedback control is the process of maintaining the performance at a local optimum in the presence of disturbances to the process such as step disturbances. Feedforward control is the modification of the recipe for a given process step based on the results measured on the same product at a previous process step. The Run by Run Controller modifies the recipe between runs, but does not modify the equipment parameters during a run. This task is left to the Real Time Controller.

The Real Time Controller accepts as inputs all in-situ measurements and modifies the recipe during a process step in order to bring the result as close as possible to target. In principle, the Real Time Controller is compensating for small differences that exist between one product run and another. The differences might be due to incoming product, incoming raw materials, or equipment operation. When these differences can be characterized before the run starts, the Run by Run Controller can be used to compensate for them. When the differences can only be characterized by in-situ measurements, the Real Time Controller must be used to compensate for them.

The Statistical Process Control module implements automated charting and interpretation of SPC charts. The primary purpose of the SPC module is to generate alerts concerning large changes in the process which indicate malfunctions of one sort or another. In addition, information about gradual changes in equipment condition might be fed back to the Run by Run controller to be used in feedforward modification of the recipe. A maintenance operation might be suggested earlier than scheduled maintenance if the accumulated gradual changes are large. For example, a clean-up maintenance is suggested if the build-up inside an LPCVD reactor is thick due to the repeated deposition process.

The variance propagation module serves the purpose of making decisions about when to send a lot on, when to rework it, and when to scrap it. This module is in effect designing the specification limits based on predictions about the probable outcome of continuing to process the lot of wafer given information about its current condition (10).

B. Implementation

The process control system of figure 2 can be viewed as part of a complete CIM system. The process control system provides an environment that allows for the easy integration of disparate pieces of equipment by implementing unit process control with consideration of preceding and following steps.

The authors envision that the process control system will be implemented on a microcomputer resident at each machine. Each microcomputer will gather data directly from processing equipment via of equipment communication interfaces such as the SECS protocol. Each computer will also communicate to a central facility which will support a common data base which can be accessed from all facilities of the CIM system.

III. Equipment Modeling

A. Definition

Equipment models play an important role in the described process control system. They are constructed in the three core modules to serve as forward running simulations of the equipment and process. Process control actions such as optimization, monitoring, diagnostics, and adjustment build on the equipment models. The more accurate the equipment models are, the more efficient these control actions are.

Figure 3 illustrates a generic equipment model with predicted outputs and two types of inputs; the inputs that we have control over and the disturbances that we have no control over. The inputs that we have control over include the equipment parameters and the wafer topography. The equipment parameters are parameters whose settings we have direct control over on the equipment, for example, temperature, pressure, and gas flow rates. Examples of the wafer topography might include feature size and previous layer's thickness. The disturbances are those inputs which are subject to unintended and undesired variations, for example, variations in the properties of incoming materials and variations of the equipment parameters themselves. The predicted outputs, for example, might include average thickness and variation of thickness down the tube in an LPCVD reactor for polysilicon.

B. Construction Approaches

The construction approaches of equipment models may be classified according to the amount of process physics included in the development (Figure 4). Traditionally, equipment models are constructed by two extremely different approaches: pure mechanistic modeling and pure empirical modeling. Several semi-empirical modeling methodologies which tend to combine the above two approaches have been also proposed recently.

Mechanistic models derived from the underlying process physics may be either closed form (analytical function) or numerical in nature (differential equation solved by numerical techniques). They have the advantages of broad applicability, good extrapolation beyond the range of experimental verification, and good prediction of process sensitivities. However, it is usually difficult and time-consuming to develop a complete mechanistic model because of unknown process physics. Therefore mechanistic models are often incomplete and the interpolation accuracy is restricted. They are also unsuitable for on-line process optimization and control because of the numerical nature which requires a lot of time to get the solutions.

Empirical models are usually polynomial regression models which are most effectively developed using design of experiments (DOE). These models have the advantages of easy development and good interpolation accuracy within a small experimental space. This modeling usually suffers from a poor interpolation accuracy within a large experimental space and a poor extrapolation accuracy beyond the range of experimental verification. This is due to the fact that the nonlinearity of the real relationship is usually significant within a large operating space and can't be approximated well by a low-order polynomial model. The nonlinearity is reduced in a smaller operating space and can be approximated better by a low-order polynomial model. The variance of the predicted value, however, greatly increases as one extrapolates from the initial experimental space.

Semi-empirical modeling tends to combine the advantages of the above two approaches; better modeling accuracy, broader applicability, and easier development. As mentioned above, it is usually difficult to develop a complete mechanistic model for a complex process. One proposed semi-empirical modeling is to incorporate one or more adjustable coefficients in the development of a mechanistic model. These adjustable coefficients which represent the uncertainty of the underlying process physics are then calibrated using DOE. In our process control system, the equipment model which is used as the basis of the Flexible Recipe Generator is constructed by this approach.

There are many other ways to construct semi-empirical models. For example, they can be developed using statistical regression techniques with physically based transformations of the primitive parameters used in the regression. The application of dimensional analysis to the transformations of the primitive parameters was proposed by Chaddha (11) and Wehrle (12). Better modeling accuracy was found when compared to the primitive parameter models. Collins of CMU (13) also introduced a methodology for the development of a semi-empirical model. A nonlinear, multi-layer regression technique combined with an analytical process model was used to develop a hybrid equipment / process model. Lin of Berkeley (14) also reported another methodology. Physically based models were first linearized and subsequently fitted using multi-stage D-Optimal experimental designs.

C. Design of Experiments

Experimental design plays an important role in the construction of equipment models. There are two primary methods of DOE; parallel and sequential. The two approaches share the common feature of varying many or all of the experimental parameters simultaneously.

In parallel DOE, all experiments are designed at the beginning and then performed. Models are constructed after all data are collected. The basic goal is to maximize the amount of information obtained with a minimum effort. Examples of parallel DOE include full and fractional factorial designs, Box-Behnken designs, central composite designs, and Taguchi's orthogonal array (1,15,16). The adjustable coefficients of the equipment model which is used as the basis of our Flexible Recipe Generator have been calibrated using parallel DOE.

Parallel DOE is widely used for the characterization of a process at a design stage, and in off-line optimization. Parallel DOE is the most appropriate approach when one must spend a great deal of time in either the preparation or running of experiments and the possibility exists to prepare or run the experiments in parallel. The classic application which spawned the development of parallel DOE is to agriculture where experiments usually require an entire growing season to complete, but many experiments may be run simultaneously. Another example would be the design of a reactor where one of the experimental variables is the geometry of a susceptor, which might take 8 weeks to fabricate. In this case, three or more susceptors could be fabricated in parallel before starting the experimental runs. However, in application to on-line optimization and control, parallel DOE presents a problem in that many of the design points produce scrap (material that is out of specification). Parallel DOE often makes scrap because of its nature of designing experiments uniformly in the entire process space. This might be acceptable at a design stage when there is little concern about making scrap, but not at a manufacturing stage when each "experiment" represents daily production.

In sequential DOE, each new measurement is used to help design the next experiment. Each new data point is used to update the models of the process and those new models are used to design the next run. The basic goal is the realization of the improvement of a process with a minimum cost. Examples of sequential DOE include Evolutionary Operation (17), the Simplex method (18), and the Ultramax method (19). In our process control system, statistical models which are used as the basis of the Run by Run Controller and the Real Time Controller are constructed from sequential DOE.

One major advantage of sequential DOE over parallel DOE is that it allows for the optimization and control of a process without making scrap. The protection against making scrap is the result of creating models which place greater emphasis on accuracy in the immediate region of operation and less emphasis on global accuracy. In addition, scrap is avoided by limiting the magnitude of the changes between runs. The ability to avoid making scrap makes sequential DOE ideally suited to on-line use.

Another advantage of sequential DOE for on-line use is that a manufacturing operation is inherently sequential. Hence, no extra delay is incurred by the use of sequential DOE as compared with parallel DOE.

IV. Flexible Recipe Generator

A. Construction Approaches

This module responds to each new product design with an indication of the appropriate recipe for the first run. The goal is to get the process into the general region of the best performance. Inputs to the Flexible Recipe Generator include mask geometry and specifications. This procedure is in one sense the reverse of many current IC process simulators. Traditionally, designers must input the process recipe for simulation and hopefully, by trial and error, meet the specifications.

Since this module is designed for the exploration of the global process space, equipment models constructed should have high accuracy in the global process space. Two alternatives are available for this purpose. One is the construction of multiple piecewise regression models, with each one simulating a separate small process space. The other one is the construction of a mechanistic or semi-empirical model which simulates the whole process space. According to the two different approaches to construct globally valid equipment models, we could propose two different approaches to construct the Flexible Recipe Generator.

The first approach corresponds to the use of multiple piecewise regression models. The Flexible Recipe Generator functions by first interrogating a set of if-then rules which select a portion of process space. The process space dictated by the if-then rules is characterized in more detail by piecewise second order regression models. The models will be used to find the best region of operation, resulting in a recipe for use in the first run of a new product.

The second approach corresponds to the use of a mechanistic or semi-empirical model. The construction of the Flexible Recipe Generator is based on the combination of an equipment model and a nonlinear optimizer as shown in figure 5. The recipe generation procedure corresponds to the optimization of an objective function which evaluates overall process performance, including consideration of rate and quality. Designed disturbances are also included in the simulation so that the obtained recipe assure that the output be less sensitive to disturbances. This idea comes from Taguchi's philosophy. We have implemented our initial Flexible Recipe Generator for the LPCVD of polysilicon based on this approach.

B. Flexible Recipe Generator for LPCVD of Polysilicon

Our Flexible Recipe Generator for the LPCVD of polysilicon builds on a semi-empirical equipment model. The equipment model is a one dimensional finite difference model which has four adjustable coefficients embedded in it representing areas of uncertainty about the process physics (20). These coefficients have been calibrated using a parallel design of experiments (Taguchi's L9 orthogonal array) in order to assure wide range of application of the model with the minimum number of experiments. The model predicts the wafer-to-wafer growth rate down the load in a tube furnace reactor (Figure 6). Inputs to the model include: silane flow rates from three injectors, injector locations, locations and temperatures of three thermocouples, operating pressure, the number of wafers, wafer diameter, the location of the wafer load, and other physical dimensions of the furnace such as tube length, and inner diameter.

In our Flexible Recipe Generator for LPCVD of polysilicon, a nonlinear optimizer called OPTDES (21) was used to find the recipe which gave the best growth rate uniformity from wafer to wafer. Each operating point suggested by OPTDES was passed to the equipment model 16 times (Figure 5), with one set of disturbances superimposed at each time. The value of an objective function is calculated according to the 16 sets of outputs and returned to OPTDES. From this information, OPTDES found another set of operating point and continued until the optimum value of the objective function was found, indicating initial recipe was generated. Taguchi's signal-to-noise ratio (SN ratio), which evaluates the ratio of mean to standard deviation, was defined as the objective function. A larger SN ratio, which means smaller standard deviation, corresponds to a better growth rate uniformity. Figure 7 illustrates the optimization results. Predicted growth rate profile which corresponds to the recipe generated by the Flexible Recipe Generator is shown as solid line. This agrees very well with the result by applying Taguchi's parameter design. Experimental result also supports the validity of the developed equipment model and Flexible Recipe Generator. Details about the comparison can be found in reference (20).

V. Run by Run Controller

A. Intent

This module starts from the recipe provided by the Flexible Recipe Generator and updates the recipe between product runs. The goal is to perform on-line optimization and control of process equipment. In response to post process and in-situ measurements, the Run by Run Controller updates its equipment models and recommends a change in the recipe in order to seek improvement of the process. This sequential adjustment procedure to seek and stay at an optimum is similar to the way a process engineer operates a piece of equipment. However, the adjustment by a process engineer is generally done on the bases of guidelines in the equipment supplier's manual and previous experience. Moreover, the control actions are limited to be single-input and single-output as more complex responses are difficult for a human to contemplate. The Run by Run Controller therefore automates and improves on the traditional actions of the process engineer by creating a mathematical formalism which allows for multiparameter responses to multiple attribute changes.

B. Sequential Design of Experiments - Overview

The algorithmic basis of the Run by Run Controller is the sequential DOE to update models for optimization and control. An early approach to sequential DOE was Evolutionary Operation (EVOP) (17). In EVOP, small parallel designs, usually limited to two or three parameters, are performed sequentially. The location of the next small parallel design is based on the results of the most recent parallel design but no results prior to that. The Simplex method is another example of sequential DOE, wherein experiments are designed one at a time. Both methods have their limitations and suffer from high risk of making scrap.

A more recent development in sequential DOE is embodied in a piece of software called Ultramax (19). In the Ultramax approach, experiments are designed one at a time, based on a weighted evaluation of all previous data. Our work on sequential DOE is built on the Ultramax method.

C. Sequential DOE in the Run by Run Controller

The use of sequential DOE for on-line optimization and control is best explained with the help of a simple example. Suppose we want to maximize a production rate (objective function) by adjusting only one equipment parameter. The underlying response of the production rate to this equipment parameter is assumed unknown in the beginning as shown in figure 8a. Note that in real situations the objective function which evaluates the overall performance is often defined as a weighted combination of outputs instead of only one output. In a more complex process, the number of equipment parameters used to optimize the objective function might be large.

The fundamental basis of the sequential DOE is to sequentially construct models after each run. Local, weighted regression,which selects and weighs the appropriate data points based on the distance from a reference point in process space and time, is adopted to ensure higher accuracy in the current region of operation. Statistical modeling is used because of its easier derivation and good adaptivity. Models constructed are equipment models which relate outputs to equipment parameters. A model which relates the objective function to equipment parameters is then constructed from these equipment models. Alternatively, the model which relates the objective function to equipment parameters can be constructed directly from the data in some cases. Suppose we have four data points at some stage in our simple example, a quadratic response surface is constructed as shown in figure 8b.

The models constructed can now be used to design the next experiment. We might design our next run based on the predicted optimum operating point; however, we usually limit the excursion for the sake of making no scrap. As one moves away from the region within which models are fitted (figure 8c), the error of fit greatly increases, representing the uncertainty of the constructed model increases and the risk of making scrap increases. One way to handle this difficulty is the superposition of a specification range onto the error of fit. We limit the excursion when projected error of fit exceeds the specification range. Another way to limit the excursion is the superposition of a constraint onto the distance travelled by the equipment parameter. The risk of making scrap is then controlled by these two constraints. The same algorithm can apply to the multiparameter cases, in which a generalized multiparameter distance is defined (22).

We continue climbing the hill as shown in figure 8d, with sequential design one at a time. Note that the adaptivity of local weighted regression, which currently creates a quadratic response surface with a concave shape in contrast to a convex shape in the beginning. After several cycles, the top of the hill is achieved, meaning that we have reached a maximum production rate.

D. Major Tasks of the Run by Run Controller

The Run by Run Controller serves the multiple purposes of local optimization, feedback control, and feedforward control.

Local optimization is the process of fine tuning a recipe given a general region in which the optimum lies. The goal is to achieve on-line optimization without making scrap. In order to test its utility for local optimization, the Run by Run Controller was applied to the LPCVD of polysilicon. In this work, the physically based equipment model developed as the basis of a Flexible Recipe Generator was used as a test bed, thus allowing for the investigation of many optimization cycles.

The optimization of growth rate uniformity from wafer to wafer within a batch was accomplished by adjusting five equipment parameters, which are the flow rates of two of the three silane injectors, the axial positions of two injectors, and tube pressure. Ultramax was used as the basis for the local optimization of the Run by Run Controller. The objective function maximized is Taguchi's SN ratio. Results of the optimization study are shown in figure 9. Figure 9a shows the run by run history of SN ratio, and demonstrates an overall trend of improvement (solid line). Solid circles which occur every three cycles correspond to optimum runs while empty circles represent exploration points which are intentionally designed a little away from optimum prediction so that we can "learn" more about the process. Figure 9b shows the growth rate profile down the length of the tube corresponding to the first run and the best run (run #64). As can be seen, the best run has a lower mean growth rate, but a substantially better uniformity, leading to a higher SN ratio. It should be noticed that we didn't start the optimization process from the recipe suggested by the Flexible Recipe Generator, but rather, from a somewhat worse recipe. This was done for the purpose of better characterization of the optimization speed of the Run by Run Controller.

Feedback control is the process of maintaining the performance at a local optimum in the presence of disturbances to the process such as shift and drift. Drift refers to gradual changes of the process over the course of time, while shift refers to relatively abrupt changes, including step disturbances. In some cases, shifts and drifts can be ascribed to known causes. For example, the build-up of deposition on the inside of a reactor can cause gradual shifting of the process. When a maintenance operation is performed, the process may undergo a sudden shift. In other cases, the cause of the drift or shift is not easily ascribed.

In order to test its utility for feedback control, the Run by Run Controller was again applied to the LPCVD of polysilicon. In this case, a step change which might result from a maintenance operation was simulated by imposing a shift in the position of two injector tubes. This simulated step change was applied between runs #64 and #65, immediately following a local optimization sequence. Figure 10 shows that the step change resulted in a process that was less well tuned, and that the Run by Run Controller was then able to respond and effect a recovery of process uniformity.

Feedforward control is the modification of the recipe for a given process step based on the results measured on the same product at a previous process step. An example of a suitable application of feedforward control is in the compensation of autodoping in epitaxy. In this process step, it is often found that dopant from a highly doped substrate out diffuses and affects the resistivity of a lightly doped epitaxial layer. In such cases, it may be possible to measure the resistivity of the incoming substrates and change the dopant gas flow rates in order to bring the resistivity of the epi layer closer to target. The Run by Run Controller can accomplish such feedforward control by constructing models for the resistivity of the epi layer which include terms for the resistivity of the incoming material.

As a general matter, it is probably advisable to exercise caution in the use of feedforward control, as the possibility exists to induce unstable interactions of two or more processes. Thus, each unit process should first be made as robust as possible, then optimized and controlled as well as possible. Only then should feedforward control be considered for application.

VI. REAL TIME CONTROLLER

The Real Time Controller accepts as inputs all in-situ measurements and modifies the recipe during a process step. The Real Time Controller has the distinguishing characteristic that it must include the time response of the system in its control algorithm. In current plans, the steady state regression models constructed by the Run by Run Controller will be combined with time constants which characterize the system to enable real time control. The time constants of the system may be expected to be relatively independent of the recipe currently in use and therefore may be characterized in a separate series of experiments. This module is still under development.

A Real Time Controller of plasma etching was recently developed by Mclaughlin at Austin (23). Steady state models were first developed to relate equipment parameters to both process parameters (in-situ measurements) and performance variables (outputs). These models were regression models constructed by parallel DOE. Step tests were run to determine process parameter time constants. Control theories were then applied on the efficient control of these process parameters to their settings by adjusting equipment parameters.

VII. RESEARCH TOPICS

A. Rapid Local Optimization

While the Run by Run controller was successful at optimizing a process as shown in figure 9, it was quite slow at doing so. We will increase the speed of response of the Run by Run Controller when used in local optimization by pursuing two approaches.

Our first approach is to pursue multiple response surfaces. The speed of response at figure 9 is limited by the fact that only one quadratic regression model which relates SN ratio to five equipment parameters was used to optimize the process. Alternatively, we can model directly the measured output, creating one model for each measurement site. Then, the quality metric, SN ratio, can be calculated from these models. The advantage of the new approach is that in the region of the optimum for the SN ratio, the individual measurements are not near an optimum. Thus, while the SN ratio must be modeled by a quadratic expression, the direct measurements can be modeled by simple linear expression. The result is a much faster optimization as there are fewer terms to calibrate from the data. We will develop multiple modeling approaches to accommodate batch to batch, wafer to wafer and within wafer uniformity data. Our initial test of this approach on LPCVD of polysilicon has shown only 15 runs are needed to achieve the optimum.

Our second approach concerns the application of dimensional analysis to equipment modeling. Through the use of the Pi Theorem of dimensional analysis, groupings of parameters may be developed which capture fundamental physical relationships. By using these grouped parameters it may be possible to achieve models which are superior in interpolation and extrapolation, leading to faster optimization. A preliminary test of this approach was successfully reported in (22).

B. Efficient Feedback Control

While the Run by Run Controller was successful at responding to a step disturbance as shown in figure 10, it was quite slow at doing so. Fundamentally, the speed of response is limited by the fact that the five parameter quadratic regression model being used to optimize the process has 21 coefficients that must be fitted from data. This can be seen from figure 10 that it took around 30 runs to re-optimize the process.

We will increase the feedback control of the Run by Run Controller by pursuing three approaches. The first two approaches are the use of multiple response surfaces and dimensional analysis discussed above. We will incorporate the use of transformed parameters and coefficients, which would be especially relevant when the cause of a process shift can be narrowed down by prior experience. For example, a maintenance operation often results in a shifted process, and experience will dictate a relatively short list of possible causes and associated variable transformations. These variable transformations will further limit the number of coefficients that must be refitted and therefore increase the speed of feedback control.

C. Interaction of Controller Functions

There are several possible interactions of the modules in the process control system. For example, there is a fundamental question of how to do SPC while performing feedback control. Another issue is how real time control affects the interpretation of data for feedback control in the Run by Run controller module.

D. Use of Statistical Models for Real Time Control

The Real Time Controller has the distinguishing characteristic that it must include in its algorithm the time response of the system. One can characterize the system time constants in a separate series of experiments and combine these with the steady state response from the statistical models. The combination is a complete system model which can be used for real time control. More algorithmic bases of the Real Time Controller need to be developed in the near future.

ACKNOWLEDGEMENTS

Research support by the DARPA under contract MDA972-88-K-008 is gratefully acknowledged.

REFERENCES

(1) G.E.P. Box, N.R. Draper, "Empirical Model-Building and Response Surfaces", John Wiley & Sons, 1987.

(2) A.I. Khuri, J.A. Cornell, "Response Surfaces", Marcel Dekker, 1987.

(3) R.H. Myers, A.I. Khuri, W.H. Carter, "Response Surface Methodology : 1966-1988", Technometrics, vol. 31, no. 2, pp. 137-157, May 1989.

(4) G. Taguchi, "Introduction to Quality Engineering", Asian Productivity Organization, 1986.

(5) M.S. Phadke, "Quality Engineering Using Robust Design", Prentice Hall, 1989.

(6) K. Dehnad, "Quality Control, Robust Design, and the Taguchi Method", Wadsworth & Brooks/Cole, 1988.

(7) R.J. Ross, "Taguchi Techniques for Quality Engineering", McGraw-Hill, 1988.

(8) D.C. Montgomery, "Introduction to Statistical Quality Control", John Wiley & Sons, 1985.

(9) H.M. Wadsworth, K.S. Stephens, A.B. Godfrey, "Modern Methods for Quality Control and Improvement", John Wiley & Sons, 1986.

(10) C.R.Shyamsundar, P.K.Mozumder, A.J. Strojwas, "Statistical Control Of VLSI Fabrication Process : A Software System", IEEE Transactions on Semiconductor Manufacturing, vol. 1, no. 2, pp. 72-82, May 1988.

(11) P.S. Chaddha, "Comparison of Equipment Modeling Methods as Applied to the LPCVD of In-situ P-doped Polysilicon", M.S. Thesis, Mechical Eng., MIT, 1989.

(12) W.P. Wehrle, "Application of Dimensional Analysis to Statistical Process Modeling", M.S. Thesis, Mechanical Eng., MIT, 1989.

(13) D.J. Collins, "A Methodology for the Development of Semiconductor Equipment Models", M.S. Thesis, Electrical and Computer Eng., CMU, 1989.

(14) K.K. Lin, J. Huang, C. Spanos, "Statistical Equipment Modeling for VLSI Manufacturing", Electrochemical Society Meeting, Fall, 1989.

(15) D.C. Montgomery, "Design and Analysis of Experiments", John Wiley & Sons, 1984.

(16) D. M. Steinberg, W. G. Hunter, "Experimental Design : Review and Comment", Technometrics, vol. 26, no. 2, pp. 71-130, May 1984.

(17) G.E. Box, N.R. Draper, "Evolutionary Operation", John Wiley & Sons, 1969.

(18) A. Adelman, W.F. Stevens, "Process Improvement by the Complex Method", AIChE Journal, vol.18, no.1, 1972.

(19) C.W. Moreno, "Self-learning Optimizing Control Software", Instrument Society of America, Robotics and Expert Systems Conference, Houston, Texas, 1986.

(20) E. Sachs, G. Prueger, "Equipment Models for Process Optimization and Control Using Smart Response Surface", Electrochemical Society Meeting, Fall, 1988.

(21) A. Parkinson, R. Balling, J. Free, "OPTDES: A Software System for Optimal Engineering Design", in Proceedings of the American Society of Mechanical Engineers Conference on Computers and Engineering, Las Vegas, August 1986.

(22) M.E. Storm, "Sequential Design Of Experiments with Physically Based Models", M.S. Thesis, Mechanical Eng., MIT, 1989.

(23) K.J. Mclaughlin, "Development of Techniques for Real Time Monitoring and Control in Plasma Etching", Ph.D Dissertation, U. of Texas at Austin, May 1989.

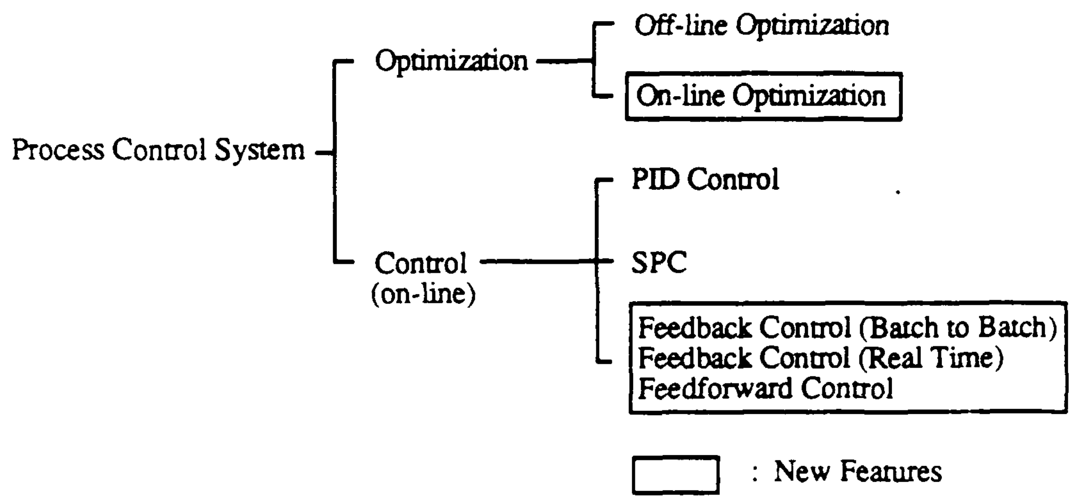

Figure 1. Major functions of a process control system

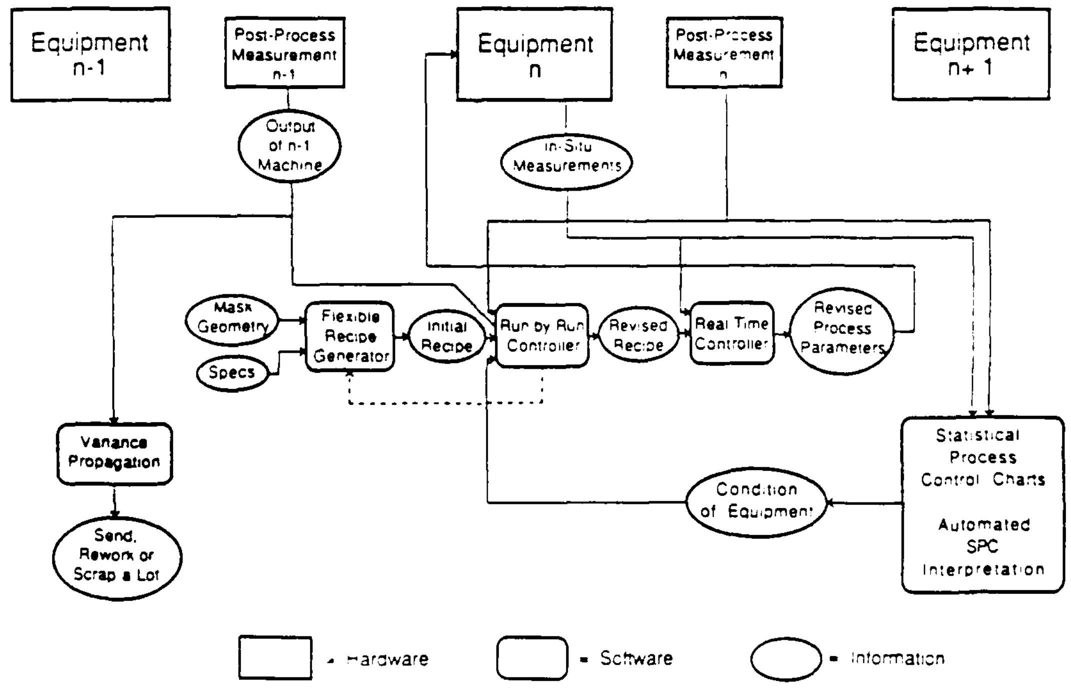

Figure 2. Block diagram of a process control system for VLSI fabrication

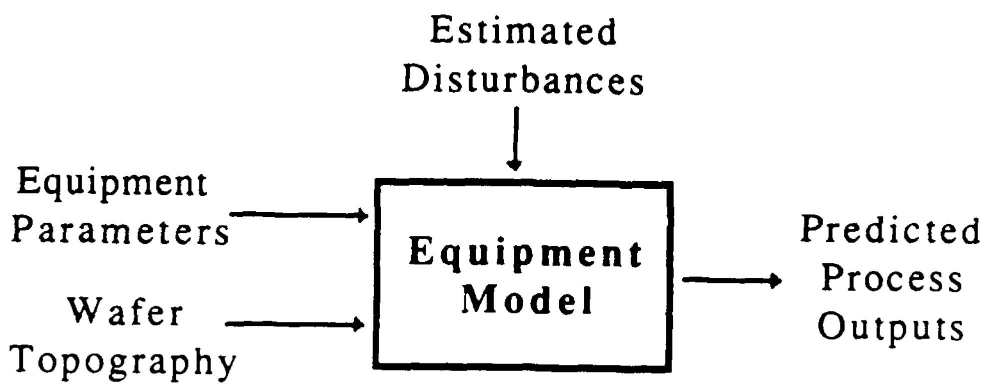

Figure 3. Definition of an equipment model

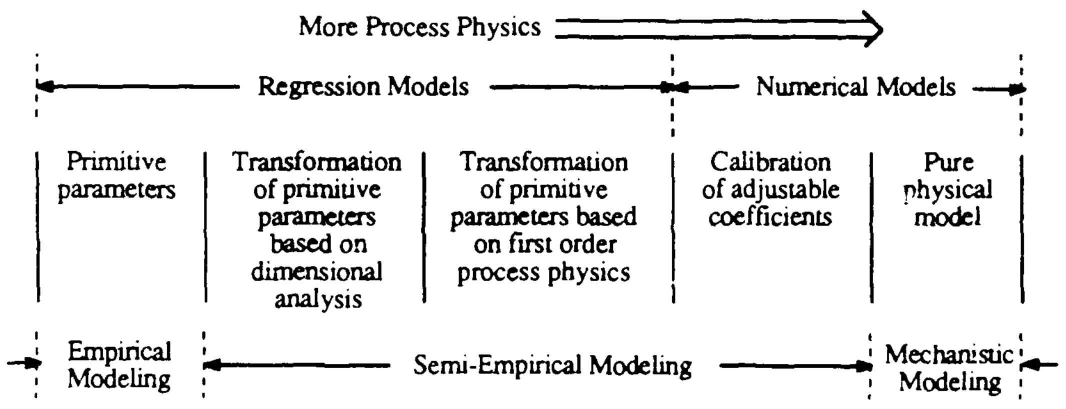
More Process Physics
Regression Models
Numerical Models
Primitive parameters
Transformation of primitive parameters based on dimensional analysis
Transformation of primitive parameters based on first order process physics
Calibration of adjustable coefficients
Pure physical model
Empirical Modeling
Semi-Empirical Modeling
Mechanistic Modeling

Figure 4. Construction approaches of equipment models

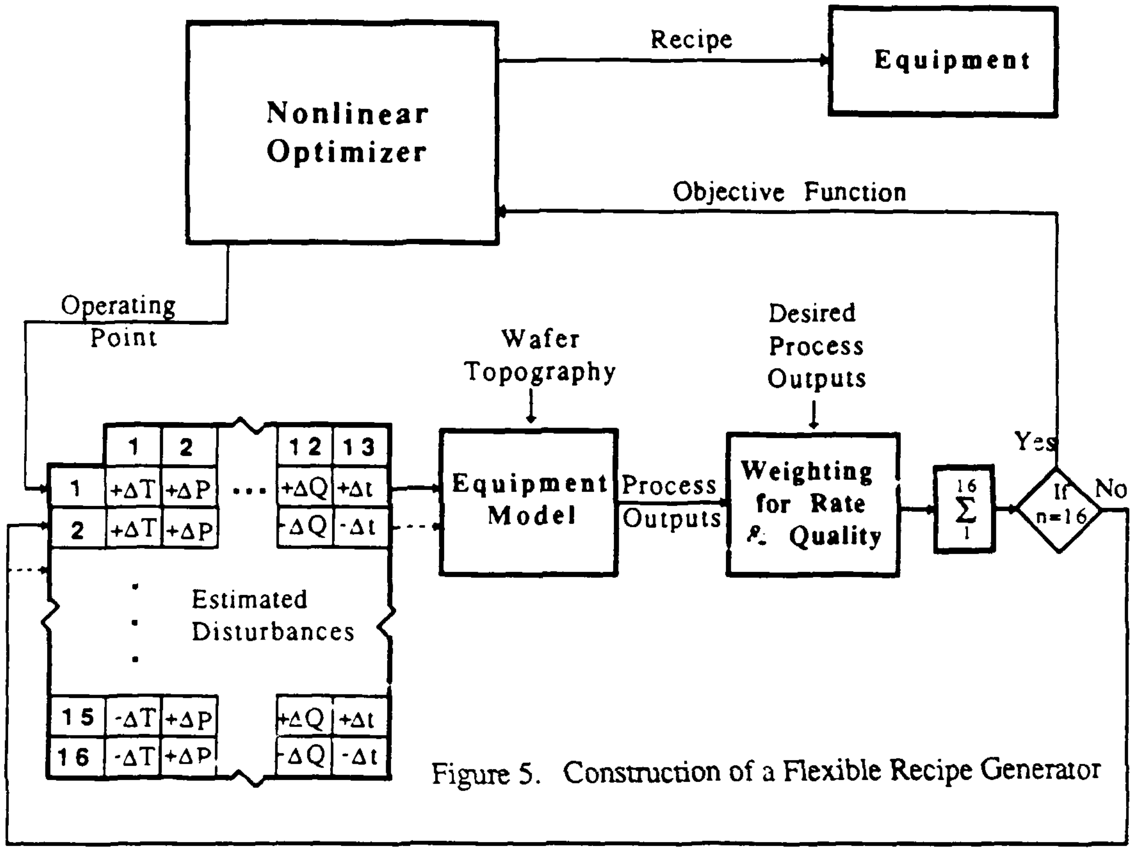

Figure 5. Construction of a Flexible Recipe Generator

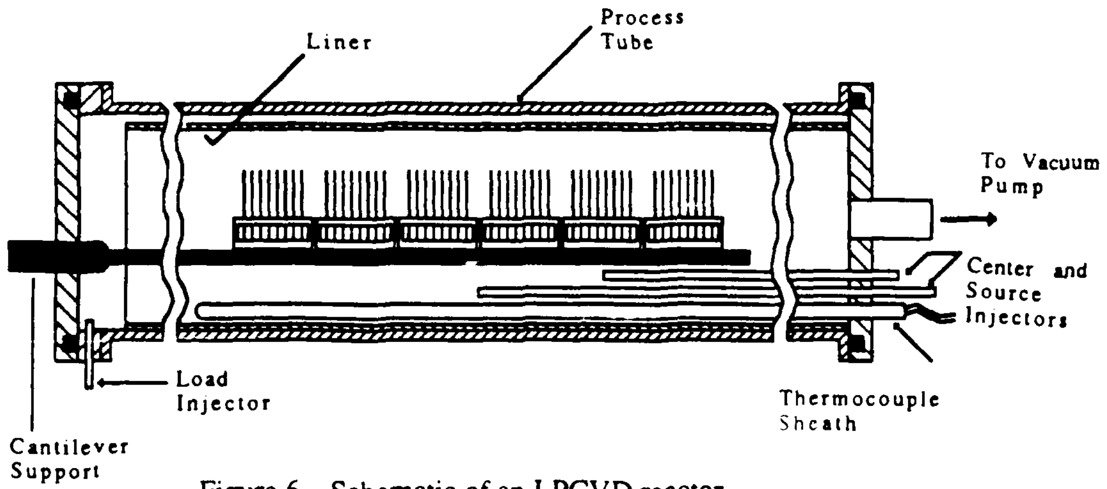

Figure 6. Schematic of an LPCVD reactor

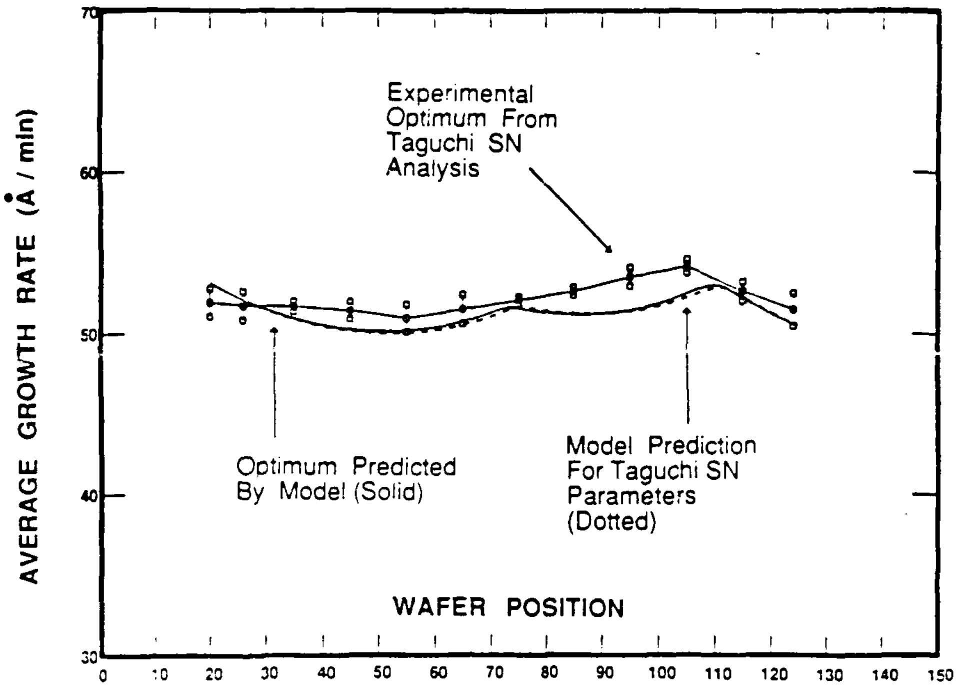

Figure 7. Optimization results of a Flexible Recipe Generator

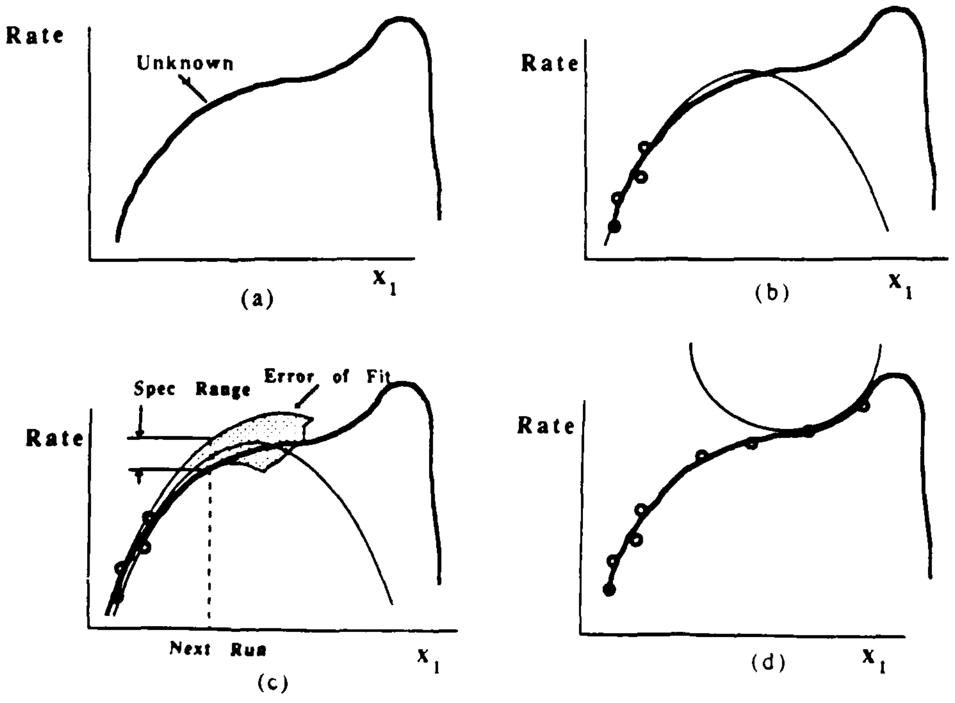

Figure 8. Sequential design of experiments

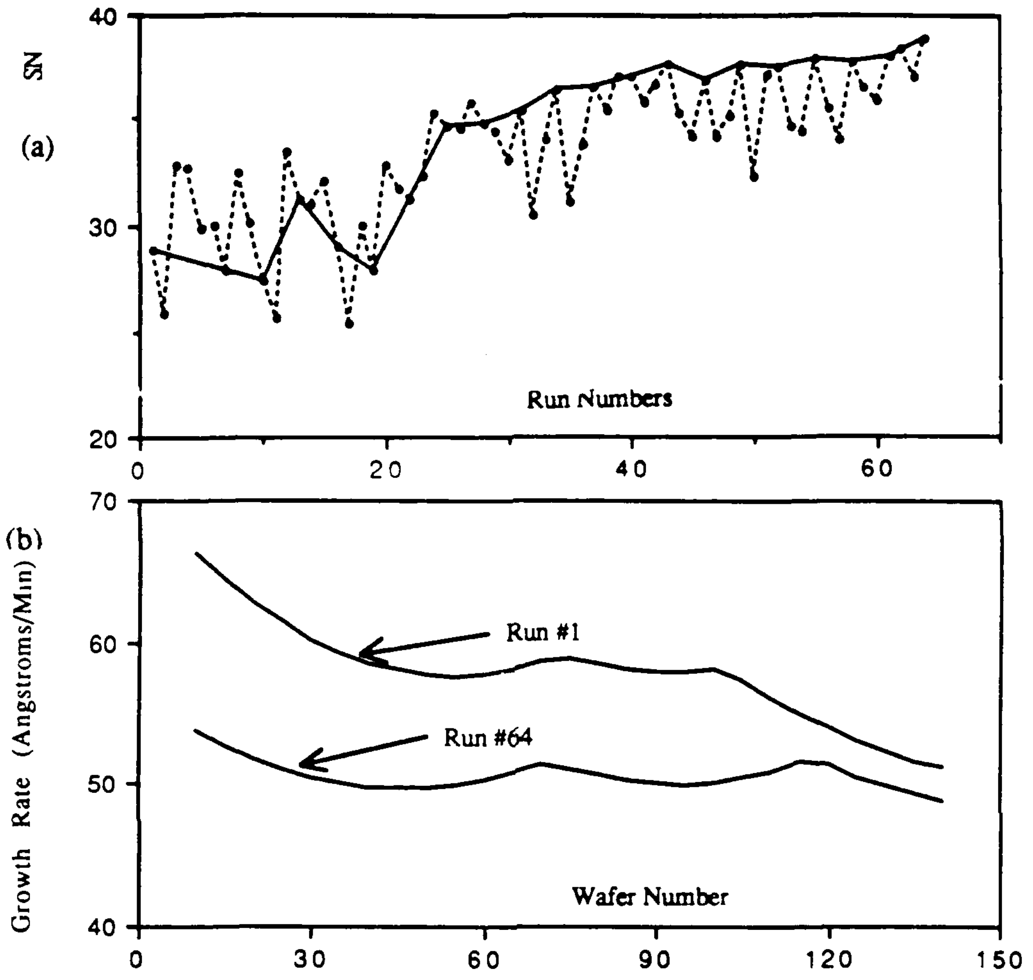

Figure 9. Local optimization of a Run by Run Controller

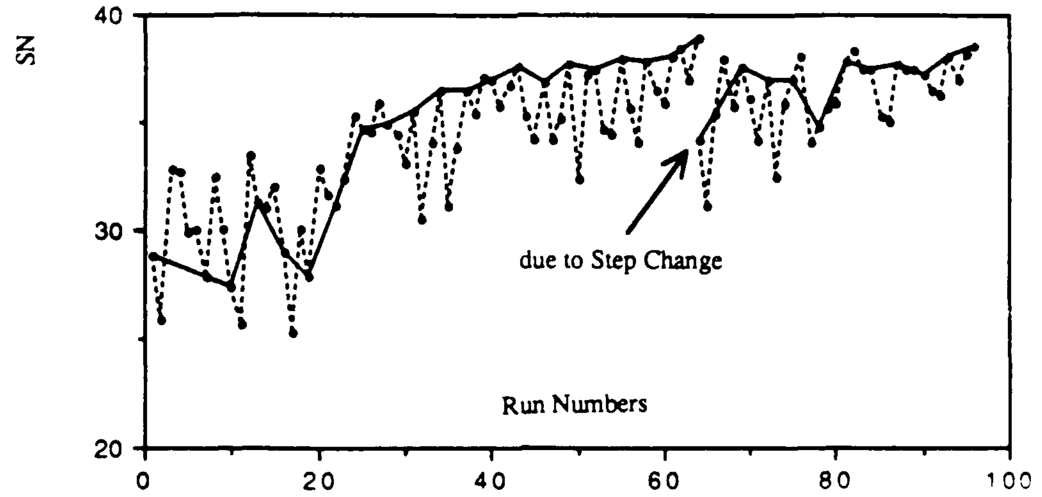

Figure 10. Feedback control of a Run by Run Controller on a step change

MASSACHUSETTS INSTITUTE OF TECHNOLOGY VLSI PUBLICATIONS

VLSI Memo No. 89-552
June 1989

Retrieving and Integrating IC Fabrication Data from Dissimilar Databases

Michael P. Ruf

Abstract

Factory personnel need to access data from many aspects of the fabrication environment. Many IC fabrication facilities store manufacturing data in distributed, heterogeneous database networks. Retrieval and integration of data can be a cumbersome task due to this configuration. The ideal solution to these problems is to standardize a data model, storing the manufacturing data in a single database. However, since such a standardization is not likely to occur in the near future, a more immediate solution is needed. This article discusses a system addressing these problems: DRIFS - *A Data Retrieval Interface for Integrated Circuit Fabrication Systems*. A uniform query interface and data model is defined for heterogeneous, distributed fabrication databases. A DRIFS prototype is described and evaluated.

Acknowledgements

This research was supported in part by the Defense Advanced Research Projects Agency under contract number MDA972-88-K-0008.

Author Information

Ruf, *current address*: Rational, 3320 Scott Blvd., Santa Clara, CA 95054-3197. (408) 496-3600.

Retrieving and Integrating IC Fabrication Data from Dissimilar Databases

by
Michael P. Ruf

Massachusetts Institute of Technology
Cambridge, MA 02139

June 7, 1989

Abstract

Factory personnel need to access data from many aspects of the fabrication environment. Many IC fabrication facilities store manufacturing data in distributed, heterogeneous database networks. Retrieval and integration of data can be a cumbersome task due to this configuration. The ideal solution to these problems is to standardize a data model, storing the manufacturing data in a single database. However, since such a standardization is not likely to occur in the near future, a more immediate solution is needed. This article discusses a system addressing these problems: DRIFS - *A Data Retrieval Interface for Integrated Circuit Fabrication Systems*. A uniform query interface and data model is defined for heterogeneous, distributed fabrication databases. A DRIFS prototype is described and evaluated.

1 Introduction

With the growing complexity of integrated circuit processes, the increasing number of technologies and product lines, and the relentless demand for higher volumes, commercial IC fabrication is increasingly dependent on information management. Information is collected from almost every aspect of the manufacturing process. Financial, marketing, design, engineering, production, scheduling, resource management, parametric, and control information all contribute to a vast collection of fabrication data. To maintain high productivity, IC manufacturers must draw on technical, historical, and corporate data at all levels of the fabrication process. More importantly, this data must be integrated and easily retrievable.

The past decade has seen the growth of commercial MIS systems for fabrication data. These systems are responsible for handling information for all aspects of the fabrication cycle, from design through packaging. However, in most cases, the systems are piecemeal, covering only a particular stage or aspect of the fabrication process. For example, separate computers and software are commonly used for logic design, process design, scheduling, fabrication, and facility support. Because these systems do not share data or communicate efficiently with each other, they do not support integration of data at a global level. Furthermore, these large conglomerations of data are not always equipped with efficient retrieval interfaces. Due to the magnitude and complexity of such systems, often only highly trained individuals can access and interpret the data. Improving the inadequacies of existing IC fabrication management systems will be a key factor in adapting to the growing information needs of factories of the future.

2 Fabrication Data in Heterogeneous Databases

The majority of the fabrication data is usually managed by a shop floor control (SFC) system. Several shop floor control (SFC) systems for IC fabrication are in use today. PROMIS[1] and COMETS[2], the two leading commercial manufacturing systems use table-based databases. SFCs normally handle a major portion of the information needs for a particular facility, but concentrate on data directly associated with the operation of the IC plant. This data can be separated into three major categories: facilities, engineering, and lot tracking/history.

Supplemental systems are used to collect and manage additional information not covered by shop floor control systems. Such systems might specialize in the following areas: testing/yield, scheduling, equipment maintenance, equipment control, financial/marketing, and facility support. The shop floor databases and the databases which store additional manufacturing data will be referred to as *local databases*. Often, users want to perform queries which draw on data from two or more of these local databases. How these queries are accomplished is an integration issue. One common method of integrating the data is to download information from supplemental databases into the main shop floor control database. This method has a number of disadvantages:

- Any outside information integrated into the SFC system must conform to the structure of the SFC database. If this structure cannot accommodate a certain data model, part of the data may be forfeited in converting to a suitable model.

- Integration is not real-time. An intermediate program must port the data from supplemental database to an SFC database. Queries may generate outdated information because the intermediate program can only be run periodically.

- As applications change, or new applications are written, integration needs may change. To restructure the integration model, the schema of the SFC databases must be modified. This is a high overhead procedure. Therefore, this integration method is not dynamic with regard to information needs.

- Because all access to integrated information must be through the SFC system, the demands on the system increase as more information from other databases is duplicated in the SFC databases to support integration.

[1]PROMIS is a registered trademark of PROMIS Systems Corporation.

[2]COMETS is a registered trademark of Consilium, Inc.

Because different people associated with the fabrication process may want access to different types of information, it is helpful to categorize these people into several groups with similar information needs. A typical breakdown is: equipment operators, engineers, manufacturing managers and supervisors, quality control personnel, production planners, support and maintenance personnel, and top level managers. Most shop floor control systems provide reports tailored for certain groups listed above, but cannot provide each group with a separate "view" of the underlying schema. Hence, new reports are difficult to generate, since retrieval interfaces are not designed to address the specialized interests of each group.

3 Multi-database Techniques

There are two prevalent approaches to integrating MDBSs. One approach is to provide a global schema which defines an integrated view of all data[1],[2],[3],[4],[5],[6],[7]. Under this configuration, the details of the local databases are hidden from the global user, who conceptually manipulates a classical homogeneous database. The second approach is to require explicit manipulation of the local databases, but provide mechanisms for sharing information between them[8],[9],[10].

An advantage of the global schema approach is that the LDBs can retain complete autonomy. Furthermore, retrieve-only integration of existing databases can be achieved without modifying or extending their schemata or functionality. Integrated update under the global schema configuration is a more complex issue, since update instructions must be applied to the global schema, presenting open problems associated with concurrency and redundancy. Several retrieve-only interfaces to MDBSs have been designed to generally address issues of the global schema approach. Among these are Multibase[1], Sirius-Delta[4], Mermaid[5], R*[6], and Gestalt[7].

Multibase, developed by the Computer Corporation of America, is centered around the ADAPLEX[3] database language, also developed by CCA. Multibase is designed to accommodate a broad range of data models in the underlying local databases. To accomplish this, it uses its own DBMS (the "Local Data Manager") to manipulate and process data from the LDBs. Sirius-Delta, a prototype system developed at INRIA, aims to allow users to manipulate existing distributed data as a unique database. It is implemented using the existing services of the LDBs. Mermaid, developed at Unisys Corporation, is intended for relational database management systems and uses existing DBMSs to manipulate and process the LDB data. R* is a prototype designed at the IBM Almaden Research Center to support transparent access to distributed relational databases through an extension of SQL. Gestalt is the central information-support system for the CAF Project (Computer-Aided Fabrication of Integrated Circuits) at M.I.T. It is designed to support read/write access to a broad range of underlying data structures, from conventional objects like personnel and machine records to unconventional objects like wafer models, IC masks, process flow programs, etc.

The integration of MDBSs can be accomplished in two stages[1],[4],[5]. First, a uniform data model and query interface are provided for retrieving data from each of the local databases. Second, an integrated representation is defined to allow access to multiple local databases via a single global query. This two-stage approach generally requires at least one new schema to be defined using the uniform data model for each of the existing LDBs. These new schemata provide homogeneity, since each of the heterogeneous LDBs are represented at this level using the same data model. Once the LDB schemata have been expressed in the uniform data model, a global schema can be built to integrate the data. The user may then submit queries in terms of the global schema, which conceptually represents an integrated homogeneous database. Executing these global queries involves solving several problems, including: query decomposition and optimization[5], query translation from the uniform data model to the individual data models of the LDBs[2], resolving semantic conflicts[1], providing sufficient concurrency control, and adequate performance.

4 A Testbed for Integrating Heterogeneous Fabrication Databases

The Texas Instruments Dallas Logic II (DLOG-II) IC fabrication facility was used as a testbed for a prototype implementation of DRIFS. At the time of the research DLOG-II was a low volume, fast turn-around manufacturing facility. A significant amount of manufacturing was dedicated to special work requests such as process development experiments. Because of the developmental nature of lot fabrication at DLOG-II, the make-up of the fabrication data emphasized engineering and parametric data rather than financial and planning data. The main computing resource at DLOG-II was a VAX 8650 running at 6 MIPS with 128 MB of memory, managing the following databases:

- **PROMIS.** The shop floor control database. By far the largest and most involved system, demanding a major portion of the computing time.
- **Engineering Test Database.** Stores test results from various integrated circuit test equipment.
- **Engineering Data Collection Database.** Stores supplemental engineering information used for generating charts and trend analysis.

Additionally, DLOG-II maintained several supplemental databases on mainframe, mini- and microcomputers. These include a facility support/control system, equipment control and particle monitoring databases. Representations of the DLOG-II PROMIS Database and the Engineering Test Database were defined in a DRIFS prototype. Since DLOG-II does not maintain a financial database, a simple mock one was created with Ingres, a relational database management system. The financial database was created on a VAX 785 running

ULTRIX[3], an implementation of UNIX[4] commonly used on DEC equipment. A separate computer and operating system were chosen for the financial database to demonstrate how DRIFS would perform in a distributed hardware and heterogeneous operating system environment.

4.1 Structure and Content of PROMIS Database

The PROMIS database consists of about 30 internally managed ISAM[5] files[11]. Most of the data they manage can be divided into three categories: facility information, engineering, and lot tracking and history.

The most extensive and significant facility information is stored in four hierarchically related tables describing production areas, work centers, equipment types and equipment units. Production areas are groups of work centers. Work centers and equipment types are groups of equipment units.

Most of the engineering data in PROMIS is associated with IC process specification. PROMIS records processing instructions defined by engineers and relays it to operators or automated fabrication equipment. PROMIS breaks down process specification into four levels of detail: device, process, recipe and operation. The device is at the top of the hierarchy, providing the most general information. Each of the next levels incorporates a greater amount of detail regarding manufacturing specifications for the device. The operation is at the bottom of the hierarchy, giving the highest level of detail. Once all the necessary modules have been defined, they are combined to from a complete manufacturing process flow.

Device records most commonly describe summary level instructions for fabricating manufacture IC's. The information in a device record can be divided into 3 major categories: administrative, parametric, and instructional. Administrative data contains descriptive information such as dates, the device record's history, personnel associated with the device, categories, etc. Parametric data consists of parameter names and values. Parameters may specify such things as lithography masks, probe tests, equipment settings, etc. They allow a general description of a device type to be tailored for specific devices. Instructional data describes the manufacturing flow for the device. Instructions can specify how to start making a device, specify a process flow, describe inventories for the device, or describe a "nested" device. All devices have an instruction of the first type, and most have a final inventory instruction. Some devices, such as purchased parts, do not have process instructions. Table 1 describes some of the fields composing the DEVC record.

PROMIS process records provide a higher level of detail than the devices which reference them. Several different devices may reference the same process. In this situation,

[3]ULTRIX is a registered trademark of Digital Equipment Corporation.

[4]UNIX is a registered trademark of AT&T.

[5]Indexed sequential access method.

Field Name	Type	Description
ACTIVFLG	Char.	Flag indicating whether this is the active version of the device.
DEVFUNCNAME	Char.	Name of the Device.
FROZEN	Byte	Flag indicating whether device can be modified.
PRODSTATUS	Char.	Status code indicating availability of device for production.
CREATEDT	Date	Date and time when this device record was created.
ACTIVEDT	Date	Date and time when this device record was made active
CHANGEDT	Date	Date and time of last modification to this device.
DESCR	Char.	Textual description of device.
INSTCOUNT	Integer	Number of instructions in the device record.
INSTTYPE	Char. Array	Instruction type (starting material, process, inventory, etc.)
INSTCOMMENT	Char. Array	Comment for instruction.
INSTPROID	Char. Array	Process ID used when instruction type is PROCESS.
INSTINVENTORYID	Char. Array	Inventory ID used when instruction type is INVENTORY.
NENGPARMS	Integer	Number of engineering parameters for this device.
NPLANPARMS	Integer	Number of planning parameters for this device
PARMNAME	Char. Array	Device parameter name.
PARMVAL	Char. Array	Device parameter value.

Table 1: Description of Selected Fields in the PROMIS Device Record.

each device applies its own set of devices parameters to the process. Different sets of processes are defined for each type of IC (e.g. CMOS, NMOS, PMOS, BIPOLAR). Processes typically have between 50 and 100 steps, each one naming a recipe to be used at that point. A recipe outlines a sequence of operations on a lot at a particular work center on a particular equipment type. Each step, or operation of a recipe describes a set of actions to be performed by equipment operators. Operations are the most detailed building blocks for processing specifications. They contain parametric and textual instructions for operating fabrication equipment.

PROMIS continually collects information on each lot as it is processed. This information, called lot data, is stored in two files: the active lot file and the lot history file. The active lot file contains one record for each lot, describing what is currently happening to the lot. Active lot records are updated at each work center. The lot history file stores an account of what happened to a lot at each s ep of processing. Each time a lot is tracked into a new work center, lot history entries are recorded.

4.2 Structure and Content of Engineering Test Database

The engineering test database (TDB) was designed and implemented by DLOG-II to manage control test data from IC test equipment. Each kind of tester at DLOG-II produces a different type and format of data. In fact, data formats can change from session to session on the same tester. The engineering test database consolidates the floating format test data from all the machines, and provides a standard query mechanism to the data.

The TDB is stored in VAX datalog format. Datalog files are composed of sequential ASCII variable length text records. There is one datalog file for each test session performed on each lot by each tester. The ASCII datalog files contain three types of records: data, header and format. There are two types of data records: test session data records, and test value data records. Test session data records contain the information necessary to locate a set of test data for any given wafer. Test value data records contain the test results for a given test session. A single test session data record precedes each set of test value data records. Header records define a format for the test session records. Format records define the format for test value data records. They specify field names which describe the values in the data records that follow. Figure 1 is an example of a datalog TDB file with header, format, test session data, and test value data records.

```
HDR1 TESTER TECH DEVICE LOT WAFER TEMP STATUS TEST-DATE TEST-TIME
FMT2 SUPPLY
FMT3 ICC2 ICC3 ICC5 VIL VIH VOH VOL
DAT1 SENTRY50 DMOS DPU-1 137380 23 25C PREBURN_IN 861231 14:49:34
DAT2 LHH
DAT3 832.0E-03 -2.000E-03 174.14E-03 1.399 1.99 1.988 338.1E-03
DAT3 821.0E-03 -2.000E-03 175.14E-03 1.320 1.98 1.981 340.2E-03
 :
```

Figure 1: Example of a TDB file.

4.3 Structure and Content of Financial Database

Because DLOG-II does not maintain a networked financial database, a small database was created using one Ingres relation. The relation stores fictional flow cost and yield data associated with each device. The relation used to store the data is composed of the following domains:

- **devid.** Device ID.
- **nobars.** Number of bars, or die, per slice.
- **cqflowcost.** Average flow cost per slice for the current quarter.
- **cqyld.** Average yield for the current quarter.
- **pqflowcost.** Average flow cost per slice for the previous quarter.
- **pqyld.** Average yield for the previous quarter.

The the values for the devid field were chosen to match the DEVNAME field[6] from selected devices in the PROMIS database. The values for the remaining fields were supplied by the modified cost data.

[6]See Table 1.

4.4 Retrieval Mechanism for PROMIS

A PROMIS module, DATALINK, allows extraction and manipulation of data from any PROMIS file. Once selected, the DATALINK menu provides general extraction functions as well as data conversion functions. The General extract function provides an interactive method of retrieving data from individual PROMIS files. The extracted data is stored in an ascii work file in the user's PROMIS directory.

After initiating the General extract function, the user must specify extraction and search criteria. The extraction criteria identify the fields in a particular file to be extracted, while the search criteria, or query constraints, specify which entries to extract. Search criteria are entered as a field name, a relational operator, and a value. Once the extraction is complete, PROMIS can convert the work file to several different formats, including the standard data interchange format (DIF).

4.5 Retrieval Mechanism for the Engineering Test Database

The engineering test database provides a program, TDBEXTR, to extract data. The extract criteria, or query constraints, can be entered from the VMS command line. TDBEXTR generates three output files: a description file, a data file, and a log file. The extract criteria are specified using a simple extract language. Each criterion begins with a field name and is followed by a list of values for that field, enclosed in parenthesis. The values can be singular, a list separated by commas, a range separated by two periods, or a wild card expression.

4.6 Retrieval Mechanism for the Financial Database

Data is retrieved from the financial database using QUEL[12], the standard query language for Ingres. Query constraints in QUEL are called clauses. Each clause consists of a pair of expressions separated by a comparison operator. Basic queries can be formulated using only constant and attribute expressions. Attributes take the form *variable.domain*, where *variable* specifies a particular relation, and *domain* identifies a field in that relation. A typical clause might consist of an attribute followed by a comparison operator followed by a constant. For example, *cost.nobars* < *200* identifies the Cost Relations in which the field *nobars* contains a value less than 200. QUEL clauses can be linked together with logical operators (and, or, not) to form a qualification. An Ingres retrieve command specifies the relation and domains to extract from each tuple, as well as a qualification indicating which tuples to retrieve. Normally, Ingres will print the results of the query on the screen, but an optional argument to the retrieve command can specify a new relation to hold the output data. Ingres also provides a copy command to port data from an ingres relation to an outside file.

5 Overview of DRIFS

DRIFS[13] is a retrieve-only interface to heterogeneous, distributed fabrication databases. It is designed using the global schema approach to provide an integrated data representation without requiring changes to the existing local databases. It provides a software environment on a separate computer system, defining and storing a representation of each fabrication database. Using those representations, it generates queries particular to each fabrication database and retrieves data from them via computer networks.

5.1 DRIFS Schema Levels

In order to separate the tasks of providing homogeneity and integration, DRIFS breaks data acquisition into three levels: the local database level, the primitive level, and the user level. The local database level consists of the shop floor control system database and the supplemental databases.

For each local database schema, the primitive level stores a translation to the DRIFS data model. In answering queries posed at the primitive level, data is transferred from the underlying local database level through various networking techniques. The primitive level provides homogeneity, because it essentially maps each data model of the heterogeneous local databases to the uniform DRIFS data model. This is an involved task, since many issues must be considered, such as computer networking, data models and query languages at the local level, and data formats of output files at the local level. While data representations from separate local databases are all stored using the DRIFS data model, they are not integrated at this level.

Once the data has been mapped to the uniform data model of the primitive level, it can be integrated at the user level. This level combines representations from the primitive level to allow data from various local databases to be integrated in common structures. User level data representations can be tailored for specific groups such as planners or engineers, allowing each group to have separate "views" into the primitive level.

5.2 DRIFS Data Structures

The DRIFS data model is expressed as *primitive structures*, which are templates for data retrieved from the local databases. Each structure consists of several titled slots where atomic data is stored. Each slot is defined with the following fields: slot name, slot type, list/singleton identifier. The slot name describes what kind of data is stored in the slot, the slot type specifies what format the data is stored in (i.e. string, number), and the list/singleton identifier indicates whether the data is stored as a list of values or as a single value. Associated with each primitive structure is information specifying from which local database to retrieve its data and how to query that database. This information indicates in which relation, file, record, etc. the data is stored. It also defines a variable map linking each slot in the primitive structure to a particular field in the local database. Primitive structures are grouped by local database, and the collection of all primitive structures for a particular local database makes up DRIFS's representation for that database.

The user level provides the means for integrating related data from each local database. User level structures combine data from the primitive structures by incorporating relevant slots from each primitive structure. The primitive structures which comprise a particular user level structure must have at least one slot in common. These key slots are used to combine the results of primitive level queries.

6 The DRIFS Prototype Implementation

DRIFS was prototyped on a Texas Instruments Explorer LISP machine. The Explorer is a single-user workstation designed for rapid development and prototyping in a symbolic processing environment. Running Common LISP with incremental garbage collection, the Explorer can manage up to 128 megabytes of virtual memory. The internal Nubus architecture uses a 32-bit LISP processor running at 10 megahertz (100 ns clock period). An ethernet controller is provided for communications with local area networks. The Explorer provides several networking services, including transparent file I/O, remote login, and task-to-task communication. To support these services, the Explorer uses the multiple communication protocols. Figure 2 shows the networking configuration for the Explorer used to prototype DRIFS. The networking names of the machines are shown in parenthesis.

6.1 DRIFS Software Environment

The DRIFS prototype was implemented using windows and pop-up menus in which primitive and user level structures are created and manipulated. To define a structure, the user specifies a DRIFS structure name, a local database, a query type for that database, and each slot in the structure. DRIFS then creates a LISP flavor as a template for data retrieved from the local databases. When data is retrieved from a local database it is parsed into primitive structures, and stored in flavor instances for later viewing.

DRIFS identifies a set of query types associated with each local database: PROMIS, engineering test, and financial. Ideally, only one query type would be needed for each database. However, since the file structure for the PROMIS database is not consistent, a separate query type was defined for reading files with multiple record types. Slots are defined by specifying a slot name, slot type, and list/singleton specifier. The slot name is used only by DRIFS as a reference and the slot type indicates the type of data which will be stored in the slot.

Once the primitive structure has been defined, the user must create a mapping which associates each slot in the structure with a specific field in the PROMIS database file. The PROMIS fields are selected using pop-up menus. Since PROMIS treats array fields

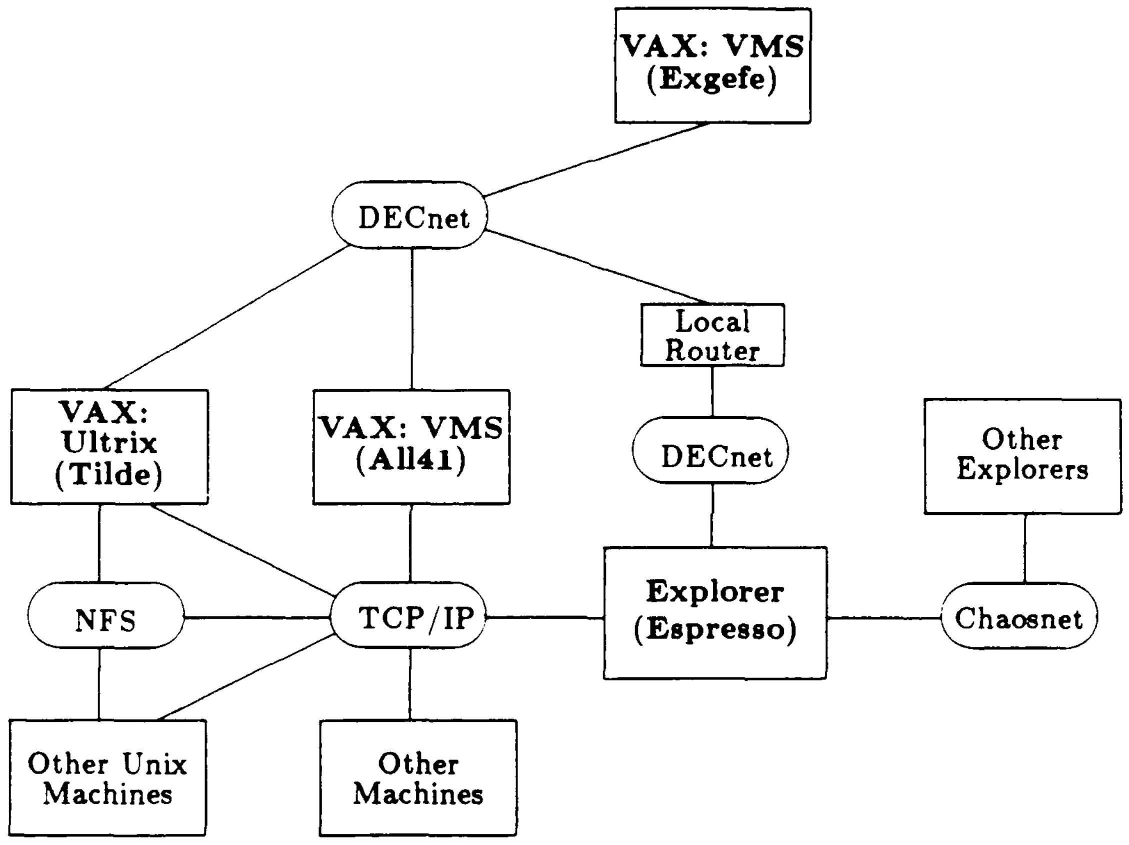

Figure 2: Partial schematic representation of Ethernet network at TI Computer Science Center

The networking name of each computer is shown in parenthesis.

differently from other fields, they must be handled specially by DRIFS. When an array field is selected in a mapping, DRIFS asks for the maximum number of elements to retrieve from the array and for the PROMIS field which specifies exactly how many elements are in the array.

User level structures combine slots from primitive structures, allowing data from separate databases to be integrated in a common data model. They are created using pop up menus that specify the component primitive structures and which slots to include from each one. One slot from each primitive structure must be identified as a key slot. DRIFS expects the data in key slots to be stored in the same format across primitive structures.

6.2 Retrieving Primitive Level Data

To initiate a DRIFS primitive level query, the user selects a primitive structure to retrieve and specifies the search criteria (known to DRIFS as query constraints) via pop-up menus. Once the query constraints have been specified by the user, DRIFS translates them into the query language of the local database and issues them to the server for that database. Two methods were evaluated for communicating query commands to local database servers: remote batch jobs and ascii-translating character streams.

Batch jobs have the advantage of standard handshaking provided by the network protocols, allowing for better handling of error conditions, and less programming overhead. However, response time for batch jobs can be unacceptably slow, depending on the configuration of the host computer. For example, EXGEFE (the host computer for PROMIS and the engineering test database at DLOG-II) gives batch jobs minimal priority at peak computing hours. In fact, during certain times of the day, batch processing virtually stops until the computing load is relieved. Another disadvantage to remote batch jobs is that they must initiate and release a new process each time a retrieval is requested. A large portion of the time it takes to run a PROMIS query is consumed in getting to the proper PROMIS menu and exiting the system. With batch jobs, this must be done each time a query is run.

In contrast, an ascii-translating character stream is a direct connection to another computer via remote login. The stream has an output buffer through which a program can send commands to the remote host as if a user were typing them. The output from the host is piped into the stream's input buffer where it can be interpreted by DRIFS. Ascii-translating character streams have the advantage of efficiency but are more prone to unanticipated errors. For example, if the host broadcasts a system message to all users, DRIFS may not know how to interpret that message, or, if a process times-out or aborts for any reason, DRIFS may continue to send commands to the non-existent process, not knowing it has been deactivated. Also, if a stream has been inactive for a certain length of time, the remote login session may terminate automatically in what is called an autologout. Although procedures may be included to detect and recover from such conditions, it is difficult to anticipate all failure modes.

Ascii-translating streams were used to communicate with the PROMIS and the engineering test databases on EXGEFE, and batch jobs were used to communicate with the financial database on TILDE. Batch jobs were acceptable for TILDE because even at high system load, there was little or no time between queuing and execution. Since direct network connections for ascii-translating streams may not exist from the retrieval computer to the local host, DRIFS defines a connection path which lists the hosts, or connection nodes, leading to the desired host. The ascii-translating character stream must login to each node separately until it reaches the desired host. The connection nodes hold the information necessary for login at each host.

6.3 Retrieving Primitive Structure Data

Note from Figure 2 that there is more than one network path between ESPRESSO and EXGEFE. To generate a retrieval connection, ESPRESSO can use DECnet expressly to connect to the local router, and then to EXGEFE, or it can use TCP/IP to connect to ALL41 and then login to EXGEFE through DECnet. Because the Explorer provides more comprehensive and reliable support for TCP/IP streams, connection nodes were defined to use the latter path.

To minimize use of the character stream, DRIFS writes its queries to a script file on EXGEFE and uses the stream only to activate and monitor the progress of the script command. Due to a problem with the Explorer implementation of DECnet, files longer than about 600 bytes could not be transferred directly from ESPRESSO to EXGEFE. To work around this problem, script files were transferred indirectly to EXGEFE via ALL41. Each script file includes commands to perform the specified query and to convert the PROMIS output file into DIF format. Using DECnet, the DIF file is then read from EXGEFE and parsed into primitive structures.[7]

DRIFS uses a separate character stream connected to EXGEFE to query the engineering test database. This character stream communicates at the VMS command level, passing query constraints to the extract program (TDBEXTR) from the command line. Using DECnet, the description file is then read to identify the format of the data file. Then, the data file is read and parsed into primitive structures.

To query data from the financial database, DRIFS creates on TILDE a script file containing Ingres commands. A batch file is used to load Ingres and activate the script, which retrieves the requested fields and copies them to an output file. DRIFS reads the output file from TILDE, parsing the data into primitive structures.

[7]The Explorer DECnet failed only when transferring files *to* EXGEFE. Therefore, DECnet could be used to transfer the DIF file directly from EXGEFE to ESPRESSO.

6.4 The User Level Interface

6.4.1 Combining PROMIS and Financial Data

To describe the DRIFS user level interface, an example user level structure, FIN-DEVICE, will be used. It combines data from two primitive level structures: DEVICE from PROMIS (Figure 3) and DEVICE-COST-DATA from the financial database (Figure 4).

FIN-DEVICE (Figure 5) incorporates fields from it component primitive structures which might be of interest to a production planner. Thus, retrieving FIN-DEVICE structures requires data to be integrated from PROMIS and the financial database.

DRIFS divides a user level query into separate primitive level queries and uses the key slots to match corresponding results from each query. For example, to view all FIN-DEVICE's with device ID's beginning in "ASAM", DRIFS would perform two separate primitive level queries. First it would retrieve device data from PROMIS by querying all DEVICE structures with NAME's beginning in "ASAM". Then it would retrieve cost data from the financial database by querying all DEVICE-COST-DATA structures with NAME's beginning in "ASAM". At this point, DRIFS compares the data in the key slots (in this case, the NAME's) to match each DEVICE structure with its corresponding DEVICE-COST-DATA structure. Once these primitive structures are paired up, each pair is combined to form a FIN-DEVICE structure.

In the preceding example, the query constraints concerned the key slots of the DEVICE and DEVICE-COST-DATA structures. If no key slot was included in the constraints, the user level query would be handled differently. For example, to retrieve all FIN-DEVICE structures with current quarter yields less than 40%, DRIFS would first retrieve all DEVICE-COST-DATA structures from the financial database with current quarter yields less than 40%. Then, it would use the data in the DEVICE-COST-DATA key slots (in this case, the NAME's) to build the query constraints for the DEVICE structure. DRIFS would use those query constraints to retrieve from PROMIS a matching DEVICE structure for each DEVICE-COST-DATA structure it has already retrieved from the financial database. The pairs are then combined to form FIN-DEVICE structures.

6.4.2 Combining PROMIS and Engineering Test Data

Integrating data from PROMIS and the engineering test database cannot be handled the manner described in Section 6.4.1, because there is not a one-to-one correspondence between PROMIS entries and entries in the test database: For each lot in PROMIS, there are several entries in the entries in the test database; for example, suppose the user wants to combine data in the ACTIVE-LOT primitive structure from PROMIS (Figure 6) and the LOT-TEST-RESULT primitive structure from the engineering test database (Figure 7). There are many LOT-TEST-RESULT's which correspond to each ACTIVE-LOT. To handle this situation, DRIFS allows user level structures to identify *sub-structures*. Substructures are primitive structures, such as LOT-TEST-RESULT, which have a many-to-one correspondence to other primitive structures, such as ACTIVE-LOT. The user level

DEVICE (PROMIS)

NAME:	string	SINGLETON
DESCRIPTION:	string	SINGLETON
ACTIVE:	string	SINGLETON
FROZEN:	number	SINGLETON
STATUS:	string	SINGLETON
CREATE-DATE:	string	SINGLETON
ACTIVE-DATE:	string	SINGLETON
LAST-MOD-DATE:	string	SINGLETON
INSTRUCT-TYPE:	string	LIST
INSTRUCT-COMMENT:	string	LIST
INSTRUCT-PROC-ID:	string	LIST
INSTRUCT-INVENT-ID:	string	LIST
NO-ENG-PARAMS:	number	SINGLETON
NO-PLANNING-PARAMS:	number	SINGLETON
PARAM-NAME:	string	LIST
PARAM-VALUE:	string	LIST

Figure 3: Description of DEVICE primitive level structure.

DEVICE-COST-DATA (FINANCIAL-DB)

NAME:	string	SINGLETON
BARS-PER-SLICE:	number	SINGLETON
CUR-QTR-FLOW-COST:	number	SINGLETON
CUR-QTR-YIELD:	number	SINGLETON
PREV-QTR-FLOW-COST:	number	SINGLETON
PREV-QTR-YIELD:	number	SINGLETON

Figure 4: Description of DEVICE-COST-DATA primitive level structure.

```
FIN-DEVICE

Slots:
                  NAME  (from DEVICE)
           DESCRIPTION  (from DEVICE)
                ACTIVE  (from DEVICE)
                FROZEN  (from DEVICE)
                STATUS  (from DEVICE)
           CREATE-DATE  (from DEVICE)
           ACTIVE-DATE  (from DEVICE)
         LAST-MOD-DATE  (from DEVICE)
        BARS-PER-SLICE  (from DEVICE-COST-DATA)
     CUR-QTR-FLOW-COST  (from DEVICE-COST-DATA)
         CUR-QTR-YIELD  (from DEVICE-COST-DATA)
    PREV-QTR-FLOW-COST  (from DEVICE-COST-DATA)
        PREV-QTR-YIELD  (from DEVICE-COST-DATA)

Key Slots:
                  NAME  (from DEVICE)
                  NAME  (from DEVICE-COST-DATA)
```

Figure 5: Description of FIN-DEVICE user level structure.

structure, ENG-ACTIVE-LOT (Figure 8), incorporates slots from ACTIVE-LOT which are of interest to an engineer. It also incorporates lot test data associated with each active lot by identifying LOT-TEST-RESULT as a sub-structure.

7 Evaluations

7.1 Evaluation of DRIFS Prototype Implementation

The DRIFS prototype is successful in providing a standard data model and a simplistic retrieval interface to data in heterogeneous fabrication databases. Because the mock financial database resides on a separate computer system from the DLOG-II VAX which stores the PROMIS and engineering test databases, the prototype demonstrates that the DRIFS concept can be implemented in a heterogeneous hardware environment. The window-oriented menu interface of the DRIFS prototype provides a flexible and convenient method for defining representations of the local fabrication databases.

The model for DRIFS primitive structures is sufficient for defining representations of each of the local fabrication databases. In addition to providing homogeneity, the primitive structures are easily integrated at the user level. The use of key slots and substructures allows primitive data from separate fabrication databases to be integrated via user level structures. Additionally, user level structures provide a means of defining specialized

ACTIVE-LOT (PROMIS)

NAME:	string	SINGLETON
COMMENT:	string	SINGLETON
DEVICE-NAME:	string	SINGLETON
CUR-PROCESS-NAME:	string	SINGLETON
CUR-RECIPE-NAME:	string	SINGLETON
CUR-PROD-AREA-NAME:	string	SINGLETON
CUR-WORKCENTER-NAME:	string	SINGLETON
CUR-EQUIP-TYPE-NAME:	string	SINGLETON
CUR-EQUIP-UNIT-NAME:	string	SINGLETON
COMPLETION-CLASS:	string	SINGLETON
STATE:	string	SINGLETON
STATE-ENTRY-TIME:	string	SINGLETON
STAGE:	string	SINGLETON
TRACKING-STAGE:	string	SINGLETON
CUR-STEP-NUMBER:	number	SINGLETON
END-STEP-NUMBER:	number	SINGLETON
EMPL-ID-TRACKIN:	string	SINGLETON
EMPL-ID-TRACKOUT:	string	SINGLETON
STEP-COMMENT:	string	SINGLETON
QUEUE-TIME:	string	SINGLETON
STEP-START-DATE:	string	SINGLETON
STEP-END-DATE:	string	SINGLETON
RELEASE-TIME:	string	SINGLETON
STEP-START-SIZE:	number	SINGLETON
MECH-STEP-YIELD:	number	SINGLETON
NO-OF-DIE-IN:	number	SINGLETON
DIE-STEP-YIELD:	number	SINGLETON
CUR-MECH-LOT-YIELD:	number	SINGLETON
CUR-EFF-DIE-YIELD:	number	SINGLETON

Figure 6: Description of ACTIVE-LOT primitive level structure.

LOT-TEST-RESULT (TEST-DB)

TESTER:	string	SINGLETON
TECHNOLOGY:	string	SINGLETON
DEVICE:	string	SINGLETON
LOT-NAME:	string	SINGLETON
TEST-DATE:	string	SINGLETON
TEST-TIME:	string	SINGLETON
TEST-TYPES:	string	LIST
TEST-RESULTS:	TEST-RESULT	LIST

Figure 7: Description of LOT-TEST-RESULT primitive level structure.

```
ENG-ACTIVE-LOT

Slots:
                  NAME  (from ACTIVE-LOT)
               COMMENT  (from ACTIVE-LOT)
           DEVICE-NAME  (from ACTIVE-LOT)
      CUR-PROCESS-NAME  (from ACTIVE-LOT)
       CUR-RECIPE-NAME  (from ACTIVE-LOT)
     CUR-PROD-AREA-NAME (from ACTIVE-LOT)
    CUR-WORKCENTER-NAME (from ACTIVE-LOT)
    CUR-EQUIP-TYPE-NAME (from ACTIVE-LOT)
    CUR-EQUIP-UNIT-NAME (from ACTIVE-LOT)
                 STATE  (from ACTIVE-LOT)
                 STAGE  (from ACTIVE-LOT)
       CUR-STEP-NUMBER  (from ACTIVE-LOT)
          STEP-COMMENT  (from ACTIVE-LOT)

Key Slots:
                  NAME  (from ACTIVE-LOT)

Sub-structures:
     LOT-TEST-RESULT  Key Slot: LOT-NAME
```

Figure 8: Description of ENG-ACTIVE-LOT user level structure.

"views" of the fabrication data for certain groups (i.e. engineers, production planners, etc.) This was demonstrated in the prototype through the ENG-ACTIVE-LOT and FIN-DEVICE user level structures, tailored for engineers and production planners, respectively.

While the Explorer LISP machine used to implement DRIFS provides an excellent environment for rapid prototyping, networking from the Explorer to other computers is slow and cumbersome. The work-around to the networking problem described in Section 6.3 adds substantial overhead to running PROMIS database queries from DRIFS. This overhead, combined with the already slow transfer rate (about 500 to 800 bytes/sec) between the Explorer and the DLOG-II VAX severely limits the performance of PROMIS queries from DRIFS. Table 2 shows the access times required to retrieve one DEVICE structure from PROMIS. Note that transferring the PROMIS script file to EXGEFE accounts for 68% of the total access time (due to the networking work-around). Since transfer time for script files is constant with respect to the number of DEVICE's retrieved from PROMIS, this percentage become less significant (15%) when a larger number (20) of DEVICE structures are retrieved (see Table 3).

	Transfer Script File to EXGEFE	Execute Script on EXGEFE	Read DIF Output File	Total Access Time
Trial 1	0:36	0:11	0:11	0:58
Trial 2	0:33	0:07	0:10	0:50
Trial 3	0:34	0:09	0:07	0:50
Trial 4	0:35	0:08	0:07	0:50
Average	**0:35**	**0:09**	**0:09**	**0:52**
Average %	**68%**	**17%**	**17%**	

Table 2: Access times (M:SS) to retrieve one DEVICE structure from PROMIS.

	Transfer Script File to EXGEFE	Execute Script on EXGEFE	Read DIF Output File	Total Access Time
Trial 1	0:38	1:46	1:55	4:19
Trial 2	0:41	2:19	2:00	5:00
Trial 3	0:40	1:46	1:59	4:25
Trial 4	0:39	1:49	1:58	4:26
Average	**0:40**	**1:55**	**1:58**	**4:33**
Average %	**15%**	**42%**	**43%**	

Table 3: Access times (M:SS) to retrieve 20 DEVICE structures from PROMIS.

Access times for queries to the engineering test and financial database are more acceptable. Table 4 shows the access times for retrieving test data for an active lot with 15

	Run TDB Extract on EXGEFE	Read TDB Output File	Total Access Time
Trial 1	0:19	0:07	0:26
Trial 2	0:17	0:10	0:27
Trial 3	0:20	0:08	0:28
Trial 4	0:22	0:08	0:30
Average	**0:20**	**0:08**	**0:28**
Average %	**71%**	**29%**	

Table 4: Access times (M:SS) to retrieve 15 LOT-TEST-RESULT structures from the TDB.

	Run TDB Extract on EXGEFE	Read TDB Output File	Total Access Time
Trial 1	0:31	0:25	0:56
Trial 2	0:27	0:27	0:54
Trial 3	0:23	0:27	0:50
Trial 4	0:22	0:33	0:55
Average	**0:26**	**0:28**	**0:54**
Average %	**48%**	**52%**	

Table 5: Access times (M:SS) to retrieve 114 LOT-TEST-RESULT structures from the TDB.

	Transfer Script File to TILDE	Execute Script on TILDE	Read Ingres Output File	Total Access Time
Trial 1	0:04	0:24	0:08	0:36
Trial 2	0:04	0:20	0:12	0:36
Trial 3	0:07	0:21	0:13	0:41
Trial 4	0:04	0:19	0:12	0:35
Average	**0:05**	**0:21**	**0:11**	**0:37**
Average %	**13%**	**57%**	**30%**	

Table 6: Access times (M:SS) to retrieve one DEVICE-COST-DATA structure from the financial database.

	Transfer Script File to TILDE	Execute Script on TILDE	Read Ingres Output File	Total Access Time
Trial 1	0:05	0:32	0:12	0:49
Trial 2	0:04	0:41	0:12	0:55
Trial 3	0:04	0:31	0:12	0:47
Trial 4	0:04	0:25	0:11	0:40
Average	**0:04**	**0:32**	**0:12**	**0:48**
Average %	**08%**	**67%**	**25%**	

Table 7: Access times (M:SS) to retrieve 10 DEVICE-COST-DATA structures from the financial database.

LOT-TEST-RESULT entries. Table 5 shows the access times for retrieving test data for an active lot with 114 LOT-TEST-RESULT entries. Notice that at 114 entries, reading the output file over the network is the dominant time factor. Table 6 shows the access times for retrieving the cost data associated with one device. Table 7 shows the access times for retrieving the cost data associated with 10 devices. The Ingres execution time increases from 21 sec. to 32 sec. (an increase of 52%) when the number of DEVICE-COST-DATA structures retrieved increases from one to 10 (an increase of 900%). These figures indicate that with small queries, a major portion of the access time is overhead in loading and executing the Ingres script file.

While DRIFS provides a standard retrieval interface and data model for heterogeneous fabrication databases, its effectiveness can be limited by the existing interfaces to the individual local databases and by networking demands. For example, the only interface to PROMIS data is through menus intended to handle interactive commands from a user console. Automated interaction with these menus is a cumbersome and error-prone method. DRIFS could be more effective if each local database provided a retrieval interface designed to handle automated data retrieval. Even without such interfaces, however, DRIFS can provide an excellent means for off line data retrieval in converting to a homogeneous fabrication database.

References

[1] J. Smith, P. Bernstin, U. Dayal, N. Goodman, T. Landers, K. Lin, and E. Wong, "Multibase–integrating hetergeneous distributed database systems," in *Proceedings of the National Computer Conference*, 1981.

[2] T. Landers and R. Rosenberg, "An overview of Multibase," in *Proceedings of International Symposium of Distributed Databases*, (Berlin, West Germany), 1982.

[3] A. Chan, U. Dayal, and S. Fox, "An Ada-compatible distributed database management system," *Proceedings of the IEEE.*, vol. 75, pp. 674–694, May 1987.

[4] A. Ferrier and C. Stangret, "Heterogeneity in the distributed database management system sirius-delta," in *Proceedings Eighth International Conference on Very Large Data Bases*, (Mexico City), Sept. 1982.

[5] M. Templeton, D. Brill, S. Dao, E. Lund, P. Ward, A. L. P.Chen, and R. MacGregor, "Mermaid—a front-end to distributed heterogeneous databases," *Proceedings of the IEEE.*, vol. 75, pp. 695–708, May 1987.

[6] B. Lindsay, "A retrospective of R*: a distributed database management system," *Proceedings of the IEEE.*, vol. 75, pp. 668–673, May 1987.

[7] M. L. Heytens and R. S. Nikhil, "GESTALT: an expressive database programming system," December 1987. To be published.

[8] B. Czejdo, M. Rusinkiewiez, and D. Embley, "An approach to schema integration and query formulation in federated database systems," in *Proceedings 3rd International Conference on Data Engineering*, (Los Angeles), Feb. 1987.

[9] D. Heimbigner and D. McLeod, "A federated architecture for information management," *ACM Transactions on Office Information Systems*, vol. 3, pp. 253–278, July 1985.

[10] W. Litwin and A. Abdellatif, "An overview of the multi-database manipulation language MDSL," *Proceedings of the IEEE.*, vol. 75, pp. 621–632, May 1987.

[11] *PROMIS Standard System Guide.* PROMIS Systems Corporation, 4.2 ed., 1987.

[12] *INGRES Reference Manual.* Relational Technology, Inc., Berkeley, CA., 3.0, vax/vms ed., 1984.

[13] M. Ruf, DRIFS: — *A Data Retrieval Interface for Integrated Circuit Fabrication Systems.* Master's thesis, Massachusetts Institute of Technology, Cambridge, MA, Jan. 1989.

Contents

MASSACHUSETTS INSTITUTE OF TECHNOLOGY VLSI PUBLICATIONS

VLSI Memo No. 89-518
April 1989

A Manufacturing Scheduler's Perspective on Semiconductor Fabrication

X. Bai and S. B. Gershwin

Abstract

In this paper, we describe the phenomena in semiconductor fabrication which are important for production scheduling. We focus our attention on collecting and understanding events, and describing related concepts, such as fabrication processes, operation sets, production machines, support machines, accessories, operation worker, support technicians, activities, objectives, etc. We do not suggest scheduling policies here; instead the goal is to characterize all the events that must be considered in developing such a policy.

Acknowledgements

This research was supported in part by the Defense Advanced Research Projects Agency under contracts N00014-85-K-0213 and MDA972-88-K-0008.

Author Information

Bai: Laboratory for Manufacturing and Productivity, Department of Mechanical Engineering, MIT, Room 35-104, Cambridge, MA 02139. (617) 253-2730.
Gershwin: Laboratory for Manufacturing and Productivity, Department of Mechanical Engineering, MIT, Room 35-331, Cambridge, MA 02139. (617) 253-2149.

VLSI#:

LMP#: 88-004

A Manufacturing Scheduler's Perspective On Semiconductor Fabrication

by

X. Bai and S. B. Gershwin

Laboratory for Manufacturing and Productivity

Massachusetts Institute of Technology

77 Massachusetts Avenue, Cambridge, MA 02139

March, 1989

A Manufacturing Scheduler's Perspective On Semiconductor Fabrication

by

X. Bai and S. B. Gershwin

Abstract

In this paper, we describe the phenomena in semiconductor fabrication which are important for production scheduling. We focus our attention on collecting and understanding events, and describing related concepts, such as fabrication processes, operation sets, production machines, support machines, accessories, operation workers, support technicians, activities, objectives, etc. We do not suggest scheduling policies here; instead the goal is to characterize all the events that must be considered in developing such a policy.

Contents

List of Figures

List of Tables

1 Introduction

The scheduling of semiconductor fabrication is a challenging problem because of the large number of operations and very long throughput times (also called cycle times). It has attracted attention from researchers, but there is still much work to be done. Leachman [1] describes a preliminary production planning framework for semiconductor industry. Bitran and Tirupati [2] develop production planning and scheduling models for wafer production facilities. Burman *et al.* [3] analyze the performance of IC manufacturing lines using operations research techniques. Chen *et al.* [4] study a queueing network model for wafer fabrication factories. Glassey and Resende [5] and Wein [6] report the impact of job-release rules on the wafer fabrication production. These papers all focus on limited issues and use highly specialized models.

To achieve effective scheduling, we need a better understanding of the IC fabrication environment. The purpose of this work is to study the phenomena by taking a closer look at the fabrication procedure from a manufacturing scheduler's viewpoint, and to build up a knowledge base for further mathematical modeling. Since semiconductor fabrication factories can be quite different from each other in the industry, we cannot exhaustively enumerate all possible events. We try to list as many as possible of the common and important events. We do not present any scheduling models or methods here. Instead, our goal is to characterize the phenomena and events that must be treated in the development of scheduling policies.

Many of the events that we described here are random, and impossible to predict precisely or control. Consequently, we suggest that the design of scheduling techniques will be concerned with stochastic models and issues related to computational speed.

Many of the phenomena described here, such as machine failures, maintenance, and the need to satisfy customers, are common to all types of manufacturing. Others are particularly important to semiconductor fabrication, and they include low or variable yields, very long lead times, machines that can hold many wafers at the same time, large number of operations, and re-entrant flow (in which parts travel in a cycle among the same machines several times).

Terminology is not standard throughout the industry. For example, "chips", "devices", "die", or "circuits" are fabricated on "wafers" or "slices". We use the terminology of the MIT Integrated Circuits Laboratory here.

2 Overview of the Semiconductor Manufacturing Procedure

The overall semiconductor manufacturing procedure can be roughly divided into six subprocedures: circuit design and mask preparation, wafer preparation, wafer fabrication (or wafer fab, or IC fab), probe test and sort, assembly, test and classify. Figure 1 illustrates the manufacturing process flow for a semiconductor firm. In terms of the product structures, the six subprocedures are described in the following.

Circuit design and mask preparation: According to marketing information and technology development, new circuits are laid out with the aid of a computer. A mask is a glass plate with a hard surface material such as chromium, chromium oxide, iron oxide, or silicon. An image is created in a mask by a pattern generator which removes material using a directed electron beam in a high vacuum.

Wafer preparation: This process begins with quartz which is refined into electronics grade silicon. The silicon is grown into cylindrical crystals three to six inches in diameter. Some newer systems use eight inch wafers and there is experimental work with twelve inch wafers. The cylinders are sliced into wafers which are then buffed, polished, and possibly doped with some impurity. These wafers are then ready for fabrication of circuits.

Wafer fabrication: A wafer fabrication procedure is performed in a wafer fab factory, which contains a number of machines and a workforce. Corresponding to different final products, a number of fabrication processes are run in a wafer fab. Each of the fabrication processes is a series of processing steps the wafers pass through during the manufacturing. Each processing step is associated with an operation set(opset) which consists of several operations in series on different machines. (The opset terminology is used only in the MIT Integrated Circuit Laboratory, but we find it useful and discuss it in detail in Section 3.) Common processes include CMOS, NMOS, and Bipolar. Other processes serve to make accelerometers, flow transducers, microphones, etc. Each of the processes creates useful three-dimensional structures, such as transistors, capacitors, resistors, and transducers, on the wafers. Each potential integrated circuit device is called a die, which consists of a number of those structures.

Wafer probe and sort: Each die on a wafer is inspected by using a wafer probe. Rejects are marked (sorted) so as to be discarded in the assembly procedure. The inventory of probed wafers is called die inventory. Wafer probe process consists of

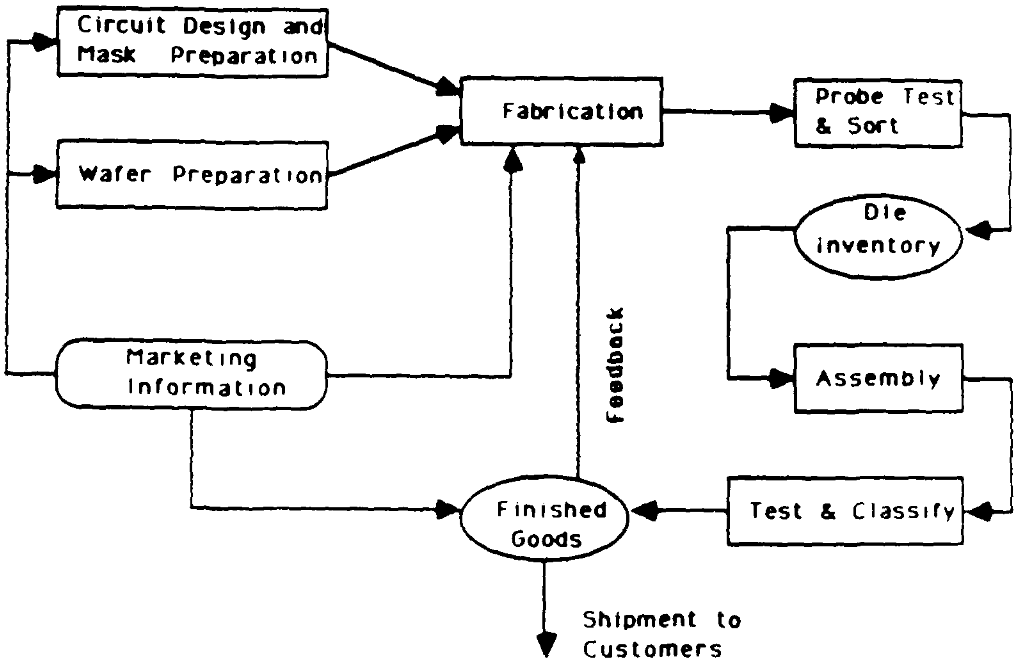

Figure 1 Semiconductor manufacturing procedure

one or only a few steps. In some firms, these steps are thought of simply as the final steps of the fabrication processes [1].

Assembly: In this process, wafers are sawed and rejected die are discarded. Good die are sealed in packages of various types [1].

Test and classify: Two test processes, raw test and final test, are involved here. In raw test (or class test) process, packaged devices are subjected to a series of tests to determine device performance. As result of this test, packaged devices are categorized into bins based on the measured performance of one or more attributes such as device speed, power consumption, tolerance of voltage variance, etc. Final tests are performed before delivering products to customers, according to the orders [1].

In this paper, we focus our attention on wafer fabrication processes to get a better understanding of the manufacturing procedure from the scheduler's viewpoint. To help describe these events, we define some terminology. A *resource* is any part of the production system that is used to perform or support fab processes. Machines, operation workers, and support technicians are resources. An *activity* is a pair of events associated with a resource. The first event corresponds to the start of the activity, and the second is the end of the activity. Only one activity can appear at a resource at any time. Operations, maintenance, and failures are activities.

In Section 3, we discuss fabrication processes, operation sets, machines, and human resources involved in IC fabrication. Activities are listed in Section 4. Constraints and objectives are discussed in Section 5 and 6, respectively.

3 A Closer Look of Semiconductor Fabrication

In a wafer fabrication factory, wafers are grouped in lots. They are grouped this way because many machines are designed to work on many wafers at the same time; because changing operations on some machines can be expensive; and because this makes it easier to trace the path of a wafer through the system for the purpose of determining causes of poor yield. Each lot of wafers travels from machine to machine, following a well-defined sequence of processing steps. We refer to the predefined sequence of processing steps as the fabrication process (or fab process) Different processes correspond to different product types. Each process consists of tens or even hundreds of processing steps (or unit processes) in series. Each processing step has an associated operation set (or opset) which consists of several

operations in sequence and information used for the operations and processing times.

3.1 Fabrication Processes and Operation Sets

Usually a wafer fab factory produces more than one product. For each product type, an operation sequence is performed to create the required structures on the wafers. *An operation is a single processing function, such as pre-oxidation clean-up or photoresist coating.* The operation sequences are called fabrication processes, such as CMOS process, NMOS process, etc. Since hundreds of operations are involved in a fab process, the operations are divided into groups, called processing steps (or unit processes), according to the processing purposes.

For example, in order to transfer a pattern from a mask to the wafers, we need to coat a photoresist layer on wafer surfaces, then expose the wafers by using an aligner, and then develop them in a wet station. After inspection, the wafers are etched and stripped in a etcher, followed by another inspection. Coating, exposing and developing have the same processing purpose: to create a patterned resist layer. consequently they are grouped together as the photo step. The etching step consists of three operations, etching, stripping and inspection.

For different patterns, we need different masks for the photo step, and for different micro-structures of the wafers, we need different etching times. That means that more information is needed to distinguish the processing steps in fabrication processes, and that is the reason to introduce operation sets (or opsets). *An opset is a group of operations which form a complete clean room processing step; it contains the information of machines, operation times (see 3.3.2), and handling times (see 3.4.1); it also specifies some parameters like furnace recipe number and photomask ID.* That is, an opset contains all the information needed to perform a processing step in a fab process.

Figure 2 depicts a simple fab process for two-mask poly gate capacitor, which consists of sixteen processing steps. Each block in the graph represents a processing step. Above the dotted line in a block is the description of the processing step and the name of the associated operation set is under the dotted line. The first processing step is field oxidation, and the associated opset is dfield5k.set, and so on. All the operations in present step must be completed before wafers go to the next step.

In the IC Laboratory of MIT, there is a baseline process, the 1.75 micron CMOS process [7], which is used to monitor equipment performance and device character-

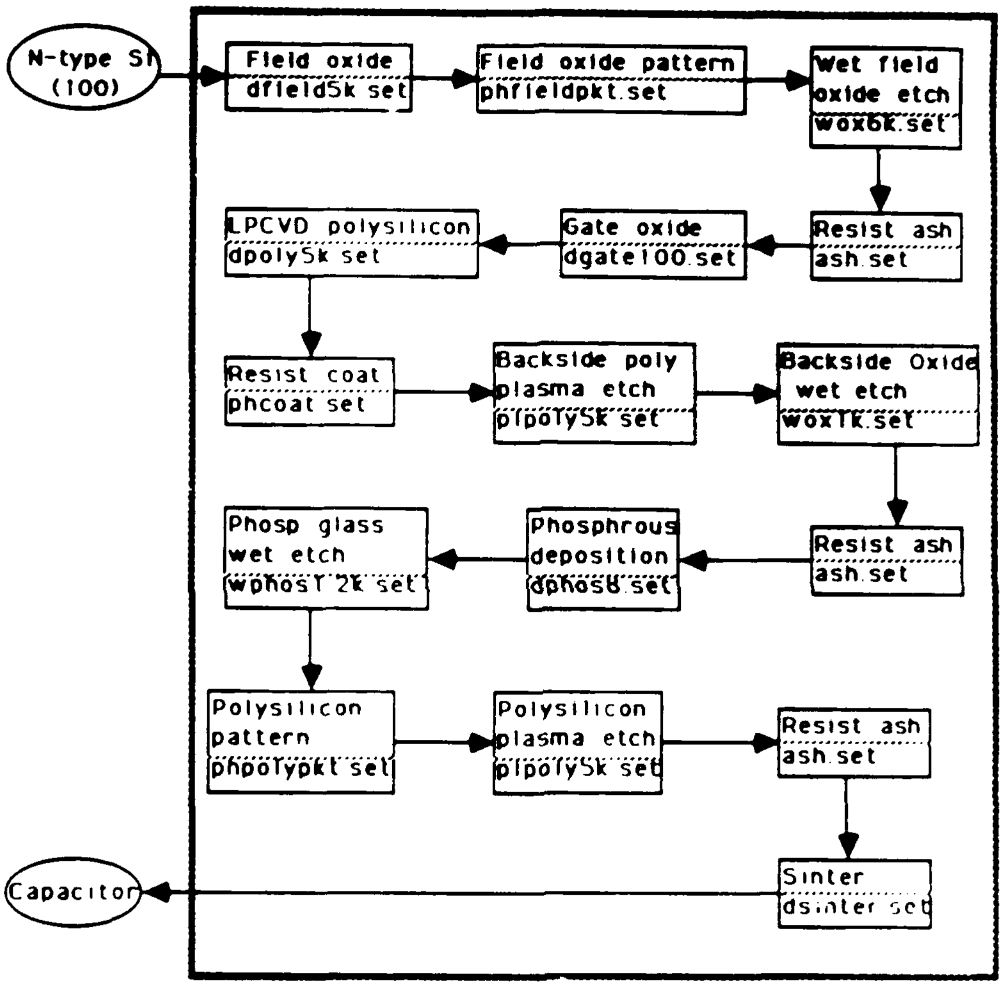

Figure 2 Two mask poly gate capacitor process

dfield5k.set

No.	operation	machine	parameter	op time	hdl time
1	RCA clean	RCA wet station		2 hrs 0 min	2 hrs 0 min
2	diffusion	furnace B1	recipe: 240 film spec avg: 5100 range: 500	7 hrs 30 min	0 hrs 30 min + 0 hrs 30 min
3	inspection	nanospec	thickness top:___ center:__ left:__ right:__ bottom:__	0 hrs 15 min	0 hrs 15 min
			total time	10 hrs 0 min	3 hrs 30 min

Table 1: An example of opset: dfield5k.set

istics. The baseline process is enhancement-compatible so that new technology innovations can be tested in a real integrated circuit process. The process was designed modularly such that coupling between processing steps (unit processes) was minimized. Whenever possible, these baseline processing steps should be incorporated into new processes and experiments. That means that all of other processes should be modified baseline processes.

Table 1 illutrates an example of opset, named dfield5k.set [8]. It consists of three operations, RCA clean, diffusion, and inspection, in sequence. The machines used for the operations are RCA wet station, furnace B1, and nanospec, respectively. The wafers undergoing this opset visit RCA wet station first, then to furnace B1, followed by inspection at nanospec. All the three operations must be completed before the wafers are ready for next opset. The required operation times (see 3.3.2) and handling times (see 3.4.1) are listed in the table. The total times include a 15 minute transportation time. The specified parameters provide further information to support each operation. For example, recipe# 240 contains the information about the temperature set-up and gas inputs for the furnace, and so on.

In terms of the processing purpose, opsets can be divided into small groups, such as diffusion, ion implantation, metal deposition, photo, plasma etch, wet etch, etc.

From this discussion, we see that each opset is like an independent micro-process, which completes a step of construction of the final device on the wafers, such as silicon oxide deposition, metal deposition, etc. The number of operations and the order of the operation sequence in a opset are usually fixed, and the operation times are fixed too.

A fab process is a sequence of opsets. A number of opsets are combined in sequence to form a process for each product type. The number of the opsets and the order of the opset sequence in a fab process may be changed to form different processes for different final products. Whenever possible the baseline processing steps should be incorporated into new processes.

There is an opset base which consists of all of the opsets involved in a wafer fab. For example, if the process in Figure 2 is the n^{th} process in a wafer fab, then the associated opset of 3^{rd} step of the n^{th} process is wox6k.set, which might be number 36 in the standard opset base.

3.2 Inspection

Most processing steps are ended by an inspection operation. The purpose of inspection is to control the product quality and to test machine performance. Decisions are made according to the inspection results. For example, good wafers which pass inspection are sent to downstream buffer to queue for next processing step, and bad wafers which fail inspection are either sent to upstream buffer for rework or scrapped to the trash can.

Not all of the production wafers go to inspection machines. Usually, only control (pilot) wafers or sample wafers are sent for inspection. When a wafer fails inspection, the cause of the failure is carefully searched, and then support technicians are notified to maintain or repair machines.

Figure 3 illustrates the inspection mechanism in wafer fabrication. Suppose we consider the opset (ni), associated with the i^{th} step of the n^{th} process, which consists of three operations,(ni1), (ni2), and (ni3). And (ni3) is an inspection operation. Wafers in lots that pass the inspection are sent to the downstream buffer B_{ni} to queue for next opset (n,i−1). When wafers fail inspection, first, we need to take one of the two choices: rework or throw them away. For example, if a polysilicon layer on the wafers is wrong, we have to scrap them, but if a silicon-nitride layer is wrong, we can send the wafers to an upstream buffer for rework. Second, if we decide to rework, we have to decide which upstream buffer to send them to, according to the

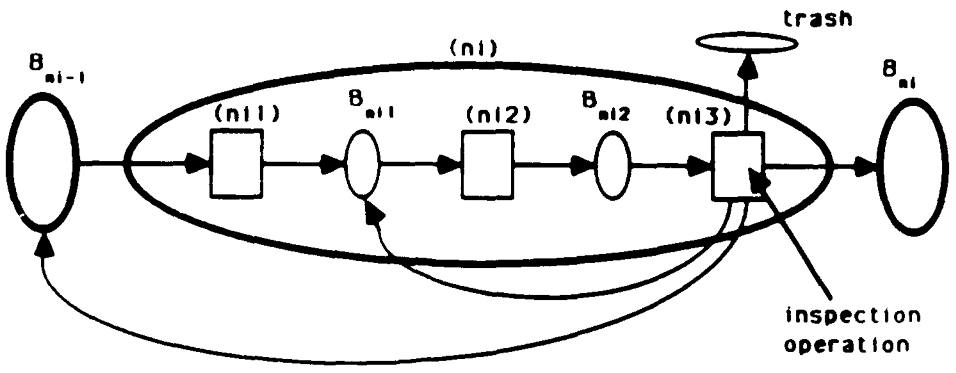

Figure 3: Inspection mechanism

inspection results. If operation (ni1) is wrong, we need to send the wafers back to the upstream buffer B_{n_i-1}, but if only operation (ni2) failed, we should send the wafers to the inner buffer B_{n_i1}. A knowledge base is needed to support this kind of decision making.

3.3 Machines

In this section we describe the machines involved in semiconductor fabrication In terms of function of the machines, we divide them into two basic groups, i e , support machines and production machines. We also discuss accessories. Machines in different wafer fab factories may not be exactly the same. The discussion here is based on the IC Laboratory of MIT.

3.3.1 Support machines

Support machines in a wafer fab are never visited by the wafers The operation states of production machines depend on the states of these machines. If one of the support machines is down or undergoing a maintenance, one or more production machines will be down In general, performing maintenance on the support machines causes a shut down of the clean room if the system is fully utilized The IC Laboratory of MIT is run only twelve hours a day. Consequently, maintenance is usually done during the off time. We refer to the whole set of support machines as the house system. Following is a list of support machines in the IC Laboratory of MIT [9].

Clean air flow: The laminar flow of filtered air is used to maintain a dust-free environment in clean rooms. This flow is provided by fans mounted on the roof, by a maze of ducts, and by filters are located above the clean rooms. Biannually, the air system is visually inspected to check the belts, fans, and bearings. The motor is lubricated; wiring is checked; control circuits, pilot lights, and interlocks are tested. Pre filters are changed quarterly, and the bag filters are changed biannually. The service takes a few days.

Emergency water: This is ordinary cold city water. It is used in emergencies to wash a person splashed by chemicals. This is biannually checked for leaks and flow testing. It may take a day or two.

Process vacuum: This is used to pick up wafers and hold wafers in place during processing. This system is serviced only when there is a problem. It has not failed since it was installed in 1985.

Cleaning vacuum: These are used to vacuum the rooms clean or clean up spills. The centrally located pumps collect waste, and the tanks must be emptied when full. The hoses are attached to the plug points in the wall from which pipes lead to the machines. This is serviced only when there is a problem. It has not failed since it was installed in 1985.

Compressed air: This can be used to operate pneumatic equipment. A dessicant at the main pumps dries the air. Biannually, the incoming filters are changed; the pumps are serviced; belts are checked; plumbing is checked; drain traps are cleaned. The maintenance duration is a few days.

Process chilled water (pcw): This is water used to control the temperature of equipment. It is recycled and reheated or recooled as required. The pump is serviced annually. Also all wiring, overloads, interlocks, control circuits, pilot lights, and voltages are checked. It is not necessary to shut down the clean room to do the maintenance. However, all related machines will be shut down when a failure occurs. This equipment was down twice at MIT during 1988, and took about one week to repair each time.

Solvent tank: A variety of chemicals are used and dumped. They are poured down the solvent waste sinks where they drain into a special underground storage tank and then are taken to long term storage. The storage tank will be emptied whenever it reaches 300 gallons. It is not currently used at the MIT ICL.

City waste: Other chemicals are neutralized to ph of 7 and dumped into the city sewer. Processing tanks are checked daily. Acid and base are added as necessary weekly. Annually, the pump is serviced, which includes lubrication and testing of wiring, voltages, and pressure differentials. The maintenance takes a few days. All wet stations are shut down when the maintenance is performed.

Fume exhaust: This is a system of ducts and fans that perform the work required to exhaust poisonous fumes from fume hoods. It includes a number of traps which should be removed and cleaned monthly. The fans are visually checked for belt tension quarterly. The system undergoes a full check of fans, annually, and all belts are changed. The maintenance may require a few days. The clean room will be shut down when maintenance is performed.

Power: Several different voltages, currents and phases are available. All of the machines require electricity to run. A backup power supply exists, so power is essentially always available. Whenever a power failure occurs, it is necessary to reset all the machines.

Deionized (DI) water: This can be considered as a storage tank with fixed volume and a production system with a maximum replenishment rate. It is possible to use up the DI water faster than it can be replenished. It is monitored daily. The DI lines require periodic cleaning which shuts the system down. The maintenance program is equivalent to about 2000 man hours yearly.

Humidity controllers and temperature controllers: Excessive moisture can damage wafers by accelerating undesirable chemical reactions on chemically active surfaces. Therefore the humidity in the clean rooms is controlled. The temperature in the clean room is controlled to within $\pm$ 1 degree because wafer dimensions depend on temperature and some equipment is highly temperature sensitive. Biannually, the bearings, dampers, and fans are lubricated. Belts are checked for tension and wear. Visually inspection of support, and vibrition check. Filter banks are also checked. Annually, motors are checked for voltage, wiring, and overloads. The maintenance may take about a week.

Tank farm: Semiconductor fabrication requires clean dry gases. At MIT, three tanks of Argon, Nitrogen and Oxygen supply the building with these gases and liquid nitrogen. Every day one percent of each of these tanks must be used or boiled off to keep the tanks cool and the temperature low. The tanks cannot be empty; they must stay partially filled to prevent contamination. Therefore operations which would empty a tank cannot be performed or the tank will be become contaminated. They are monitored daily and serviced as necessary by outside vendor.

Local gases: These are small tanks of gases placed in cabinets near the equipment which use them. They include Silane, Phosphine, Hydrogen chloride, Ammonia, Boron trichloride, Dichlorosilane, Sulfer Hexafloride, Freon 13, freon 14, freon 116, Helium, Phosphorous Pentafloride, Silicon Tetrafloride, and Boron trifloride. There are a number of gas cylinders in each cabinet. Each gas cylinder is connected through valves, to filters and then to wafer processing equipment. They are maintained annually.

Local gas vent: To ensure that the leakage of a gas cylinder does not poison anyone, the air from the gas cabinets is exhausted.

Safety alarms: Fire and gas leaks are reported by safety alarms. Fire extinquisher and fire pumps are checked annually. Hydrogen monitors and toxic gas monitors are checked daily visually and serviced monthly. Sensors are cleaned; lamps are aligned; and tapes are changed.

3.3.2 Production machines

Wafers visit the production machines and occupy them for certain periods of time. These production machines impose capacity constraints on the production rates. That is, no machine can be busy more than 100% of the time. The period of time required by a machine to perform an operation on the wafers is called *operation time*. Whenever we talk about an operation time, we must specify both the machine and the operation. For instance, in Table 1, furnace B1 needs 7 hours and 30 minutes to perform the diffusion operation in dfield5k.set. Some of this is load/unload time. In terms of purposes of the operations done by the machines, we name different groups of the production machines as diffusion equipment, lithographic equipment, inspection equipment, etc. Following is a list of production machines in MIT ICL.

Lithographic equipment:

Photolithographic operations transfer the image on a mask to the photoresist layer on a wafer. They are done in a wafer track which consists of several machines listed as follows:

HMDS vacuum bake vapor prime and image reversal system (Model 3 10): Wafers are sprayed with a dehydrating chemical, HMDS, at 150° C. A dedicated commercial oven is used for this operation, which requires house vacuum, power, and a dry and particle-free environment. It is expected to fail about once every 5 months and will require about 3 days to repair.

Photoresist coater & developer (GCA 1006 Wafertrack): After HMDS, wafers are loaded on the GCA wafer coating track for photoresist coating. The pre-exposure bake is done in the in-line contact oven module. After the exposure, the exposed wafers are developed on the GCA developing track. Post-development hard baking is done in a in-line oven. This equipment is expected to fail about once every 2 months and will require 3 days to repair.

Wafer stepper system (GCA 4800 DSW): This equipment does the exposure. The pattern transfer from an appropriate mask is carried out in a GCA 4800 , 10X direct step-on wafer system equipped with a 10-78-45 g-line lens. All the relevant information about stepping a given mask pattern on a wafer are stored in a job specification file in a dedicated PDP-11. Operation times are longer for wafers with smaller widths, due to the greater precision required for alignment. The mean-time-to-fail (MTTF) is about 2 weeks and the mean-time-to-repair (MTTR) is about 2 days.

Asher: In all baseline steps, resist is removed by washing in a photoresist stripper (Drytek Model Megstrip 6). The MTTF is about 2 months and the MTTR is about 3 days.

Etching equipment:

Dry etching: This is done in a plasma. Due to an applied voltage the ions in the plasma bombard the silicon target perpendicular to the surface. The surface is eaten away where it is not covered by resist. There are three plasma etchers in MIT ICL. Nitride etching and polysilicon etching are done in etcher-1 (LAM 480). SF6 and CCl4 are used as etching gas respectively. Etcher-2 (LAM 594) is used for oxide etching with CHF3+CF4 as etching gas. Metal etching is done in etcher-3 (LAM 690). These machines are expected to fail about once every 2 months and will require about 2 days to repair.

Wet etching (wet chemical process station): In addition to the dry etching steps, stripping of oxide and silicon nitride is done using wet chemistries. The wafer is placed in a bath, and the etch eats away at the parts of the layer not covered by resist. The MTTF is about 5 months and the MTTR is about 2 days.

Diffusion equipment:

RCA clean: Before wafers are loaded into furnaces, they are cleaned in a wet station using chemistries. It is expected to fail about once every 3 months and will require about 2 days to repair.

Oxidation furnaces: The MIT ICL is equipped with BTU oxidation furnaces. These machines are used to expose wafers to hot gases at a variety of pressures for oxidation or diffusion. Furnaces consist of quartz tubes, gas controllers, temperature controllers, a suspended loading system, and a dedicated PDP-11. The ICL staff, who is responsible for diffusion, maintains these furnaces with respect to cleanliness and routine chemical vapor monitoring. These machines are expected to fail about once a year and require 3 days to repair.

Chemical vapor deposition (CVD): Layers can be deposited on wafers, in furnaces, from gases by a process called chemical vapor deposition (CVD). If the process occurs at low pressure, it is called low pressure chemical vapor deposition (LPCVD). There are four LPCVD tubes in MIT ICL. Deposition layers include polysilicon, silicon oxide, silicon nitride, and boro-phospho-siliocate-glass (BPSG). The MTTF is about 2 months and the MTTR is about 3 days.

Metalization equipment:

Sputtering system (CVC 601): Al 1%Si is the baseline first level metal. It is deposited in a CVC sputtering system in the dc megnetron sputtering mode. The MTTF is about one month and the MTTR is about 4 days.

Ion implantation equipment:

Ion implanter: In MIT ICL, ion implantation is done in an ETON medium current machine (model NV 3206). This machine injects ions into wafers. It is operated by a special technician and requires high voltage and high vacuum. It is expected to fail about once a month and will need about 3 days to repair.

Rapid thermal annealer: Ion implantation damages the crystaline structure of the wafers. After ion implantation, it might be desirable to anneal the wafers. This may be accomplished by a rapid annealer (AG 210T-02). This machine did not fail at MIT during 1986-1989.

Inspection equipment:

To ensure that the process is within tolerances and that quality is maintained, wafers are inspected by measuring certain features, such as lithographic line width, impurity concentrations, film thickness, and electrical characteristics.

Microscope: This is used for inspection in opsets like photo and etching. The MTTF is about 2 years and the MTTR is about 5 days.

Surface profiler (DEKTAK IIA): A surface profiler is essentially a phonograph needle that measures the height of the bump that it crossses. Layer thickness is measured by first carving away valleys with a lithographic process, and then measuring the height of the remaining layer. It is expected to fail about once a year and to require 5 days to repair.

Ellipsometer (GSC L116BL-26A): This measures the reflection of a laser beam off the measured surface. This machine did not fail during 1986-1989.

Junction sectioner (PIC 2015D): This machine carves a groove in the wafer, stains the wafer, and then a microscope is used to identify the depth of the dopant. This machine did not fail during 1986-1989.

Automatic four point probe: This measures the resistivity of the incoming silicon wafers and of the layers grown on it. The MTTF is about 1 year and the MTTR is about 5 days.

CV-plotter (MDC CSM-16): This machine measures the space charge capacitance as a function of reverse bias voltage on a junction. This machine did not fail during 1986-1989.

Film thickness measurement system (Nanospec/AFT 010-0180): This equipment measures the thickness of thin film on wafers. This equipment is expected to fail about once every 1.5 years and to require 3 days to repair.

3.3.3 Accessories

In a wafer fab, some consumable materials are needed to run the fab processes. For instance, clean room gowns are worn over clothing; sticky mats at the door pull dirt off shoes; beakers are needed for some operations; tools are needed for certain maintenance and activities; chemical solvents are needed for some operations; poly gloves are needed in the clean room, and so on.

Different strategies can be used for accessories in scheduling. If we assume that accessories are always available, then we can ignore these factors in mathematical models. However, accessories are only available if they have been ordered in sufficient quantity. Disruptions are possible, but their frequencies and durations depend more on how well the fab is managed than on any physical phenomena.

3.4 Human resources involved in IC fabrication

In a wafer fab, a workforce is needed to run the fab processes. Technicians do operations and maintain equipment. Process engineers are responsible for process design and monitoring machine performance and device characteristics. A manager is in charge of smoothing processes in a whole factory. According to responsibilities, we group them as operation workers and support technicians.

3.4.1 Operation workers

This group of people is in charge of operations. They either carry wafers from one machine to another or are assigned to a certain machine and spend periods of time there to perform operations. As resources, they impose capacity constraints on production rates. That is, no person can be more than 100% busy performing operations. The period of time required by an operation worker to do an operation on a machine is called *handling time*. Depending on experience, different people need different handling times to do the same operation. We could choose the average time (or average plus one standard deviation) as the constant handling time and ask for new worker training to keep the variance of handling time as small as possible.

Whenever we talk about handling time, we should also specify the operation and the machine. For example, in Table 1, an operation worker needs 1 hour to do a diffusion operation on furnace B1, i.e., 30 minutes for loading wafers and set up temperature and 30 minutes for unloading wafers. Notice that the handling time is different from the operation time. That means between loading and unloading one machine, the worker can do operations on other machines.

In the wafer fab industry, operation workers are usually assigned to a machine. They might only know how to turn the machine on or off, load or unload wafers, and press buttons to start or end operations. In a research laboratory, operation workers carry wafers around to perform operations. They are usually more knowledgeable and more actively involved in process design and product development. In the MIT IC laboratory, about 100 graduate students are involved in wafer fabrication and most of them are in the operation worker category. Using students as workers causes scheduling complexity because of their complex personal schedules.

3.4.2 Support technicians

These people do not perform operations, and so they do not impose capacity constraints on production rates directly. But they do impose capacity constraints on both of production machines and support machines. That is, no person can be more than 100% busy maintaining or repairing equipment. This group includes technicians who maintain equipment, process engineers, managers, and other non-operation workers.

3.5 Support relationships in wafer fabrication

Figure 4 illustrates the support relationships in wafer fabrication. In a typical wafer fabrication factory, several final product types are produced. Each product type requires a number of opsets in sequence, from the opset base, to form a fab process. The arrows from the opset base to the fab processes represent the support relations. Production machines and operation workers are called from corresponding bases to support the operations. Production machines are based on support machines, and support technicians are required by both production and support machines. It should be noticed that for a certain operation, the machine is determined in the opset, but the operation worker is not.

4 Events and Activities in IC Fabrication

In this section, we discuss the activities that occur in a wafer fab factory. In terms of degree of control, we categorize activities as *controllable activities*, *uncontrollable and predictable activities*, and *uncontrollable and unpredictable activities*.

4.1 Controllable activities

This kind of activity can be arranged by a decision-maker or a manager of a wafer fab factory. Usually we can only decide when to start an activity and cannot change the time that an activity requires. But there are exceptions. For example, suppose a regular maintenance of a machine takes 3 hours for one support technician, but if two technicians work together, it may take only 2 hours. An ideal scheduling algorithm should include this kind of trade-off.

Production operations: The major activities in a IC fab are production operations, such as wafer processing, wafer inspection, and so on. These activities are well studied and accurately timed. Operation times are sometimes random. For instance, the probe time depends on the yield and the inspection time depends on the skill of the worker. For simplification, the operation times are usually treated as constants by using estimation.

Set-ups: To convert a production machine from one operation to another may require cleaning the machine, changing chemistries, or performing adjustments. All such activities are called set-up changes. Wafers cannot be processed during set-up changes, and only a restricted set of operations may be performed on the machine

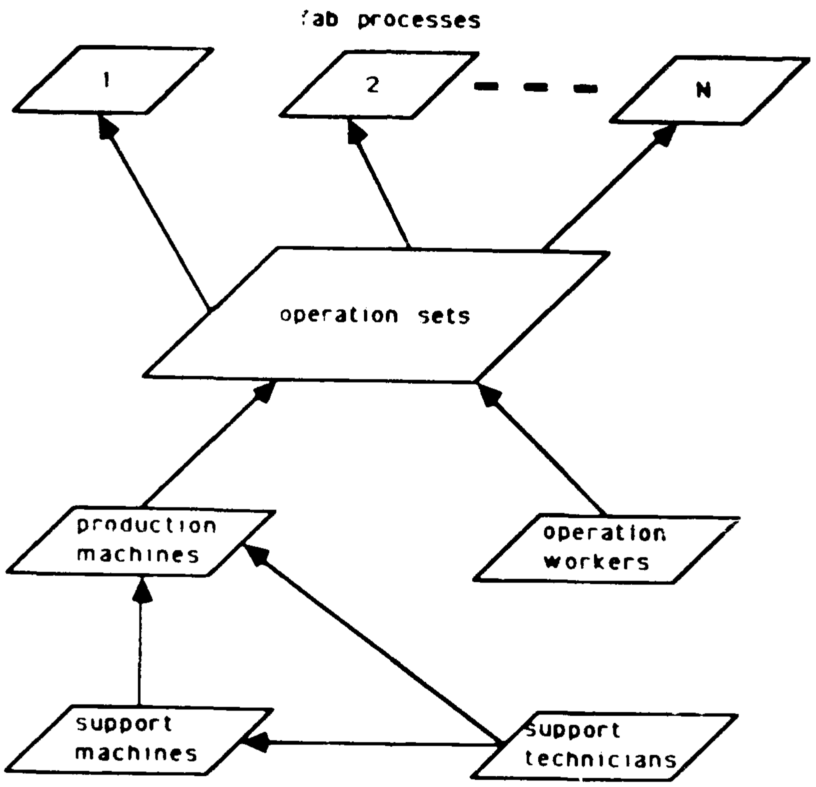

Figure 4 Support relationships in wafer fabrication

between set-up changes. Most machines, but not all, in a wafer fab factory are flexible enough to neglect the set-up change time. One of the machines for which set-up time is important is the wet-etching station for changing the chemistries.

Multiple routings: Sometimes, two machines can perform the same operation, and we have to decide which to use. Furthermore, one of the two machines, the primary machine, may be favored over the other due to speed or reliability. The secondary machine is sometimes used, but it may require a non-trivial change of process plan.

Preventative maintenance: Some regular procedures must be performed to maintain both production and support machines. If maintenance is performed late, yield may decrease, or a machine failure may occur. If maintenance is performed exactly on time it may conflict with the need to process a high priority lot. If maintenance is performed early, machine time and technician time may be wasted. Maintenance times are usually treated as constants.

Specified events for human resources: Both operation workers and support technicians are subject to some specified events. For example, new worker training, technician training, group meetings, overtime, and vacations are all activities that scheduling should be concerned about. We can decide where, when, and who to perform these activities, and we also can decide how long they will take.

Environment control: We need to do some tests regularly to ensure that yield does not suffer from variability in consumables, particle counts, humidity, machine performance, and other critical parameters. If a problem is detected, a repair may be required.

4.2 Uncontrollable and predictable activities

This kind of activities cannot be arranged by decision-makers or managers, but they know when they will happen and how long they will take.

Holidays: All human resources take holidays off. We must account for these events in advance.

Working hours: Some of the operation workers and support technicians are subject to special time schedules. For example, in the MIT IC fab laboratory, most of the operation workers are graduate students. Each of them has a specified class schedule. We cannot change the class schedule, but we may know when and how long they will be available

4.3 Uncontrollable and unpredictable activities

For these activities, we do not know either when they will happen or how long they will take. The only thing we can do is to respond as quickly as possible. But we may have statistical data on uncontrollable and unpredictable activities for long term planning.

Machine failure and repair: Both production and support machines are subject to random failures and need random repair times. In general, a machine failure may be either detected during a inspection, or anticipated due to noises the machine makes prior to failing. Once a machine failure is known, the problem can be diagnosed and a repair scheduled. To some extent, we can reduce repair time by using more technicians.

Random absence of human resources: Both operation workers and support technicians are subject to random absences due to illness, accident, etc.

Defective wafers: At various points in the fab process, entire wafers are discarded, either because the wafers failed inspection or because they are broken.

Rework: Sometimes, one or more operations can be redone when a wafer fails inspection. There are a variety of rework policies in industry. For example, in some wafer fabs, the parent lot is held until the rework wafer catches up and then the whole lot is moved to the downstream machine. However, in other factories, the parent lot moves downstream without the rework wafer.

The rework decision on a wafer is based on some empirical rules. For instance, polysilicon deposition can not be redone while the nitride deposition can be. However, if the reason for the inspection failure of a polysilicon deposition is that the layer is thinner than the specification, the wafer can go back to the tube immediately to push the layer thicker.

Engineering holds on lots: Sometimes, certain process will be stopped until some experiment results are obtained or engineering decisions are made.

Yield fluctuation: There are two yield measurements in a wafer fab. The *process yield* is the ratio of the number of wafers which reach the probe test to the original number of wafers at the process starting point. It is random since the number of defective wafers is random. The *probe yield* is the ratio of the number of good chips to the total number of chips on a wafer. The probe yield changes randomly. The yield fluctuation affects the demand to the total number of wafers which are released to wafer fab floor.

Product ratings: Some devices, such as microprocessors and memory, can be delivered with different ratings for some characteristic, such as speed. These characteristics are measured at the probe test stage. A given production process may not produce devices at a single speed; rather, it produces them with a random mix of different speeds. The mix differs for different lots, and this complicates the satisfaction of customer orders [1].

Demand change: The production demand is a function of the customer orders and production yield and usually varies randomly. In practice, demand is often treated as constant during a short planning horizon. It is not easy to estimate the future demand since so many part types are involved in wafer fabrication. Customer cancellations are a constant hazard.

5 Constraints on the Scheduling

Semiconductor fabrication require machines, people, accessories, wafers, time, and so on. Each of the requirements imposes constraints on the scheduling. Here we assume that circuit design and mask preparation, wafer preparation, and process design have been done before we implement fab processes.

Production machine capacity: Production machines process a certain number of wafers at a time. No machine can be busy more than 100% of the time performing operations or occupied by other activities.

Support machine availability: Support machines do not impose capacity constraints on production rates directly. But if cne support machine is down, one or more production machines may be down. That means that the availability of support machines can affect the production machine capacity.

Down-time delay: For some support machines, if they go down, one or more production machines will be down after a time delay. For example, if the clean air system is down, the number of dust particles will increase in the clean room. When the number reaches a certain value, all the machines which need the laminar flow of the clean air will be prevented from working.

Operation worker capacity: Each person takes a certain amount of time to complete an operation, and each person is limited to operations he knows how to perform. There are a limited nu. ıer of people available.

Support technician capacity: Each technician takes a certain amount of time to maintain or repair a machine. The support technician availability affects the repair times of both production and support machines, and therefore system capacity.

Operation sequence: In a fab process, operations have to be performed one after another following a pre-defined sequence.

Buffer size: Due to technology limitations, wafers must pass through several operations before being inspected. If an operation produces defective wafers, and there is much inventory between operations, many wafers will be produced before the faulty operation is discovered. Therefore, at some points of fab processes, buffer sizes must be small. Note that the buffer size is not necessarily the physical size. Since there always is plenty of floor space to put wafers in a wafer fab factory, we might set a threshold for each buffer to control the work-in-process inventory. When the number of wafers in a buffer reaches the threshold, we stop (or block) the upstream machine.

Limited chamber size: The number of wafers a machine can process simultaneously is limited. For instance, the diffusion tubes in the MIT ICL can process 100 wafers in a single operation. A scheduler must sometimes decide whether to process an incomplete batch or to wait for more wafers arrive.

Limited waiting time: At some points in a fab process, wafers cannot wait for a long time in a buffer, because exposing the wafer surface in the air will decrease yield. For example, after RCA clean, diffusion operations must be done as soon as possible without letting the wafers wait in a buffer for a long time. Short waiting time implies small buffer size.

Shifts: Often wafer fab factories are run one shift per day. Wafers must be in a certain state at the end of a shift. Therefore, some operations cannot be started late in a shift, because they will not be completed during that shift.

Pilot wafer runs: A pilot wafer is often run through a series of processing steps before running the whole lot. By grouping compatible lots, the scheduler can increase capacity by having lots share pilot wafer runs.

Return to same machine: In order to increase yields, it is sometimes best to restrict a lot to a particular machine on all its visits.

Product mix: Sometimes, the demands for several products are in proportion. This is the case, for example, when a line is used to make the circuits for a large computer system. It is important to keep production close to the required proportions because any deviation results in useless inventory.

The floor control system: Many existing floor control systems in wafer fabs are old and can only handle limited amount of information. A complex algorithm may not be implementable.

The multiple departments involved in managing a line: In wafer fabs, more than one departments are involved in managing a production line. One department is responsible for wafer starts; another for dispatching lots within the system; another for establishing delivery schedules to customers.

6 Scheduling Objectives and Factory Types

In this section we discuss the objectives that scheduling of wafer fab should achieve. Usually we cannot achieve all the objectives at the same time. According to production purpose, wafer fab factories are categorized as job shop type, industry type, and mixed type.

6.1 Scheduling objectives

Following is a list of objectives for wafer fab scheduling.

Inventory: Whenever possible, we want to reduce work-in-process inventory in a wafer fab factory, because it takes up space, costs money for handling, and increases throughput time. However, too little work-in-process inventory will reduce production rates.

Throughput time: This is the time that a wafer spends in a fab. It is also called *cycle time* or *lead time*. The shorter the throughput time is, the faster the system can respond to customer orders, and the faster the firm can develop new products and processes.

Variability in throughput time: Because of the many random phenomena described here (particularly yield fluctuations) the throughput time - and therefore delivery time - can be random. Large uncertainties are undesirable.

Market demand: Demand is random due to market uncertainties, and due to customer cancellations. Inventory is necessary to reduce the impact of new demands, but it is a consequence of demand decreases. An important objective is to meet varying demands as close as possible with low inventory.

Costs: Wafer fabrication is usually very costly. For instance, some equipment depreciates at \$100 per minute; and in addition to facility costs, just to maintain operating environment can cost \$3 million or more per year. Whenever possible, we want to reduce the costs. Effective scheduling can reduce material costs, resource costs, energy costs, and so on.

6.2 Factory types

Some objectives are in conflict. For example, if we want to deliver products on time, we have to build up inventory, and if we want to reduce throughput time, we have to reduce inventory. Thus, sometimes we have to make a choice of objectives for scheduling. Based on the production purpose, categorizing factory types may help make this choice.

Job shop type: In some wafer fab factories, there are many product types. The production demand is usually small for each product type, and changes randomly and rapidly. For example, in a research laboratory, we only process a small number of wafers for each experiment, and the requirements are unpredictable. In a wafer fab factory, the personalization processes, which create customized circuits, involve thousands of chip part types, and highly variable demands. We categorize this kind of factory as a job shop.

Industry type: In this kind of factory, there are few product types. Production demand is large for each product type, and is relatively predictable.

Mixed type: In most wafer fabs, both the high volume and the personalization processes are performed simultaneously. Sometimes, engineers need to do experiments in a industry type factory to develop new processes and products. Two kinds of production demands are then mixed together. Such wafer fabs are categorized as mixed type.

7 Summary

In this paper, we have discussed many of the events associated with wafer fabrication. We have attempted to describe all the issues which are important for production scheduling. The purpose of this work has been to list and understand the events so that schedulers can be designed to account for these phenomena.

8 Acknowledgment

The authors are grateful to many people both inside and outside of MIT for helpful discussions and suggestions. In particular, we thank Paul Maciel, the manager of MIT ICL, Mike Schroth, the research specialist of MIT ICL, and Prof. C. G. Sodini for their thoughtful suggestions and comments. We would like to thank C. Lozinski for providing us the unpublished material [9]. We thank L. P. Devaney, the research specialist of MIT ICL, for providing us the preventative maintenance information about the house system of MIT ICL. We also thank S. Hood and D. Connors of IBM, J. Hogge and Y. Choong of Texas Instruments, and W. J. Trybula of General Electronics for their comments and suggestions. This work was supported by the Defense Advanced Research Projects Agency and monitored by ONR under contract N00014-85-k-0213.

References

[1] R. C. Leachman, "Preliminary Design and Development of A Corporate-Level Production Planning System for the Semiconductor Industry," Tech. Rep., ORC#:86-11, University of California at Berkeley, 1986.

[2] G. R. Bitran and D. Tirupati, "Planning and Scheduling for Epitaxial Wafer Production Facilities," *Operations Research, Vol. 36, No. 1*, 1988.

[3] D. Y. Burman, F. J. Gurrola-Gal, A. Nozari, S. Sathaye, and J. P. Sitarik, "Performance Analysis Techniques for IC Manufacturing Lines," *AT&T Technical Journal, Vol. 65, No. 4*, 1986.

[4] H. Chen, M. Harrison, A. Mandelbaum, A. Van Ackere, and L. Wein, "Empirical Evaluation of A Queueing Network Model for Semicondctuor Wafer Fabrication," *Operations Research, Vol. 36, No. 2*, 1988.

[5] C. R. Glassey and M. G. C. Resende, "Close-Loop Job Release for VLSI Circuit Manufacturing," Tech. Rep., ORC#:87-8a, University of California at Berkeley, 1986.

[6] L. M. Wein, "Scheduling Semiconductor Wafer Fabrication," 1987. Unpublished manuscript, Department of Operations Research, Stanford University.

[7] P. K. Tedrow and C. G. Sodini, "MIT Twin-Well CMOS Process," 1988. Unpulished MIT ICL document.

[8] "Operation Sets," 1987. Unpublished MIT ICL document.

[9] C. Lozinski, "A Scheduling Perspective on Semiconductor Wafer Fabrication," 1987. Unpublished manuscript, University of California at Berkeley.